19981800

CONTROL DATA
CORPORATION

CONTROL DATA
CYBER 170 MODELS 172/173/174

CENTRAL PROCESSOR UNIT

THEORY OF OPERATION
DIAGRAMS

Volume 1 of 3

HARDWARE MAINTENANCE MANUAL

REVISION RECORD

REVISION	DESCRIPTION
01	Preliminary release.
(4-75)	
02	Update to reflect S/N 101 approved ECO level at 6/27/75 as per Engineering Change Order 1445.
(7-75)	
03	Update to reflect ECO 1364 (No change to this manual)
(9-75)	
04	Update to reflect ECO 1353 (pak placement change)
(9-75)	
05	Update to reflect ECO 1355 (Wire List change).
(9-75)	
06	Update to reflect ECO 1386
(9-75)	
07	Update to reflect ECO 1385.
(9-75)	
08	Update to reflect ECO 1411.
(9-75)	
09	Update to reflect ECO 1394.
(9-75)	
10	Update to reflect ECO 1404.
(9-75)	
11	Update to reflect ECO 1366 (D.P.D. 2-16)
(10-75)	
12	Update to reflect ECO 1419.
(11-75)	
13	Update to reflect ECO 1436.
(11-75)	
14	Update to reflect ECO 1478.
(12-75)	
15	Update to reflect ECO 1428.
(12-75)	
16	Update to reflect ECO 1381 (Change to D.P.D. 2-13)
(12-75)	
17	Update to reflect ECO 1480.
(12-75	
18	Update to reflect ECO 1400.
(12-75)	
Publication No. 19981800	

REVISION LETTERS I, O, Q AND X ARE NOT USED

Address comments concerning this manual to:
Control Data Corporation
Publications and Graphics Division
4201 North Lexington Avenue
St. Paul, Minnesota 55112

or use Comment Sheet in the back of this manual.

REVISION RECORD (CONT'D)

REVISION	DESCRIPTION
19	Update to reflect ECO/FCO PD 01481.
(4-76)	
20	Update to reflect ECO/FCO PD 01502.
(4-76)	
21	Update to reflect ECO/FCO PD 01516.
(4-76)	
22	Update to reflect ECO/FCO PD 01512.
(4-76)	
23	Update to reflect ECO/FCO PD 01479.
(4-76)	
24	Update to reflect ECO/FCO PD 01413.
(4-76)	
25	Update to reflect ECO/FCO PD 01518.
(4-76)	
26	Update to reflect ECO/FCO PD 01616.
(4-76)	
-27	Update to reflect ECO/FCO PD 01602.
(4-76)	
28	Update to reflect ECO/FCO PD 01601.
(4-76)	
A	Manual released.
(4-76)	
B	Update to reflect ECO/FCO PD 01420. List of effective pages added.
(7-76)	Pages affected: iia, iii, iv, v, vi, 5-1-5, 5-1-9, 5-1-11, 5-1-13,
	5-1-15, 5-2-3, 5-2-7, 5-2-11, 5-2-13, 5-2-15, 5-2-17, 5-2-29,
	5-2-31, 5-2-33, 5-2-37, 5-2-41, 5-2-43, 5-2-45, 5-2-57, 5-2-67,
	5-2-75, 5-2-93, 5-2-95, Comment Sheet.
C	Update to reflect ECO PD01640 (no textual change).
(8-76)	Pages affected: iia, iv, v, vi, Comment Sheet.
D	Update to reflect ECO PD01485.
(8-76)	Pages affected: iia, iv, v, vi, 5-1-4, 5-1-5, 5-1-11, 5-1-14, 5-1-15,
	5-2-3, 5-2-5, 5-2-17, 5-2-19, 5-2-23, 5-2-33, 5-2-51, 5-2-53,
	5-2-55, Comment Sheet.
E	Update to reflect ECO PD01703.
(9-76)	Pages affected: iia, iv, v, vi, Comment Sheet. (No textual change).
F	Update to reflect ECO PD01704.
(9-76)	Pages affected: iia, iv, v, vi, 5-2-37, Comment Sheet
G	Update to reflect ECO PD01741.
(9-76)	Pages affected: iia, iv, v, vi, 5-1-11, 5-2-29, 5-2-41, 5-2-93, 5-2-95, Comment Sheet.
H	Update to reflect ECO PD 01748.
(11-76)	Pages affected: ii-a, iv, v, vi, viii, 5-2-37, Comment Sheet.
J	Update to reflect ECO PD01772 and to incorporate Model B differences.
(12-76)	Pages affected: ii-a, iv, v, vi, vii, 5-1-13, 5-1-15, 5-2-94, 5-2-95, Comment Sheet.

Publication No.
19981800

REVISION RECORD (CONT'D)

REVISION	DESCRIPTION
K	Update to reflect ECO PD01780/CA37569. Pages affected: 11-b, iv, v, vi, 5-2-1, 5-2-3,
(12-76)	5-2-5, 5-2-7, 5-2-9, 5-2-11, 5-2-13, 5-2-37, 5-2-67, 5-2-69, 5-2-75, 5-2-81, 5-2-91,
	Comment Sheet.
L	Update to reflect FCO37692. Pages affected: ii, ii-b, iv, v/vi, 5-1-6, 5-1-9, 5-1-13, 5-2-3,
(3-30-77)	5-2-29, 5-2-41, 5-2-47, 5-2-73, Comment Sheet.

Publication No.
19981800

MANUAL TO EQUIPMENT LEVEL CORRELATION SHEET

This sheet shows the latest manual revisions, ECOs/FCOs that have affected and been included in the revisions, and equipment level (series codes) covered by the revisions.

SHEET 1 OF 2

		EQUIPMENT					
MANUAL REV	FCO OR ECO	AA107	AD103	AT253			
01	Prelim.	A02	A01	A01			
02							
03 04 05	PD01353 PD01355	A03					
06	PD01386	A05	A03				
07	PD01385	A06					
08	PD01411	A07	A04				
09	PD1394	A04	A02				
10	PD01404	A08 A12	A05				
11	PD01366	A13 A14	A06				
12	PD01419	A15	A07				
13	PD01436	A16	A08				
14	PD01478	A17 A18	A09				
15	PD1428	A19	A10				
16	PD01381	A23	A11	A02			
17	PD01480	A24 A26	A12				
18	PD01400	A27 A28	A13				
19	PD01481	A29	A14				
20	PD01402	A30					
21	PD01516	A31 A34					
22	PD01512	A35 A44					
23	PD01479	A45 A47					

MANUAL TO EQUIPMENT LEVEL CORRELATION SHEET (Cont'd)

SHEET 2 OF 2

MANUAL REV	FCO OR ECO	AA107	AA131	AD103	AD105	AT253	
		EQUIPMENTS					
24	PD01413	A48 A49		A14-A22			
25	PD01508	A50 A60		A23-A25			
26	PD01616	A61		A26			
27	PD01602	A62 A66		A27			
28	PD01601	A67		A28			
A	Released	A67					
B	PD01420	A68-A73		A29-A32			
C	PD01640	A74-A80		A33, A34			
D	PD01485	A81, A82		A35			
E	PD01703			A36			
F	PD01704			A37			
G	PD01741			A38, A39			
H	PD01748	A83-A85		A40, A41			
J	PD01772	A86	B01, B02	A42	B01		
K	PD01780	A87	B03	A43	B02		
	CA36643	A88	B03	A44	B02		
L	CA37692	A89	B03	A44	B02		

LIST OF EFFECTIVE PAGES

New features, as well as changes, deletions, and additions to information in this manual, are indicated by bars in the margins or by a dot near the page number if the entire page is affected. A bar by the page number indicates pagination rather than content has changed.

PAGE	REV	PAGE	REV	PAGE	REV	PAGE	REV	PAGE	REV
Front Cover	-	5-2-18.1	A	5-2-46.6	A	5-2-78.0	A		
Title Page	-	5-2-18.2	A	5-2-46.7	A	5-2-78.1	A		
ii	L	5-2-18.3	A	5-2-46.8	A	5-2-78.2	A		
ii-a	J	5-2-18.4	A	5-2-47	L	5-2-78.3	A		
ii-b	L	5-2-19	D	5-2-48.1	A	5-2-78.4	A		
iii	B	5-2-20.1	A	5-2-48.2	A	5-2-78.5	A		
iv	L	5-2-20.2	A	5-2-48.3	A	5-2-78.6	A		
v	L	5-2-21	A	5-2-48.4	A	5-2-78.7	A		
vi	L	5-2-22.1	A	5-2-49	A	5-2-78.8	A		
vii	J	5-2-22.2	A	5-2-50.0	A	5-2-78.9	A		
viii	H	5-2-23	D	5-2-50.1	A	5-2-78.10	A		
ix	A	5-2-24.1	A	5-2-50.2	A	5-2-78.11	A		
x	A	5-2-24.2	A	5-2-50.3	A	5-2-78.12	A		
xi	A	5-2-25	A	5-2-50.4	A	5-2-78.13	A		
xii	A	5-2-26	A	5-2-50.5	A	5-2-78.14	A		
5-1-1	A	5-2-27	A	5-2-50.6	A	5-2-78.15	A		
5-1-2	A	5-2-28	A	5-2-50.7	A	5-2-78.16	A		
5-1-3	A	5-2-29	L	5-2-50.8	A	5-2-79	A		
5-1-4	D	5-2-30.1	A	5-2-50.9	A	5-2-81	K		
5-1-5	D	5-2-30.2	A	5-2-50.10	A	5-2-83	A		
5-1-6	L	5-2-31	B	5-2-50.11	A	5-2-84.0	A		
5-1-7	A	5-2-32	A	5-2-51	D	5-2-84.1	A		
5-1-8	A	5-2-33	D	5-2-53	D	5-2-84.2	A		
5-1-9	L	5-2-34	A	5-2-55	D	5-2-84.3	A		
5-1-10	A	5-2-35	A	5-2-56.0	A	5-2-84.4	A		
5-1-11	G	5-2-36	A	5-2-56.1	A	5-2-85	A		
5-1-12	A	5-2-37	K	5-2-56.2	A	5-2-86.0	A		
5-1-13	L	5-2-38.1	A	5-2-57	B	5-2-86.1	A		
5-1-14	D	5-2-38.2	A	5-2-58.1	A	5-2-86.2	A		
5-1-15	J	5-2-38.3	A	5-2-58.2	A	5-2-87	A		
5-1-16	A	5-2-38.4	A	5-2-59	A	5-2-88.0	A		
5-2-0.1	A	5-2-39	A	5-2-60.0	A	5-2-88.1	A		
5-2-0.2	A	5-2-40.0	A	5-2-60.1	A	5-2-88.2	A		
5-2-1	K	5-2-40.1	A	5-2-60.2	A	5-2-88.3	A		
5-2-2.1	A	5-2-40.2	A	5-2-61	A	5-2-88.4	A		
5-2-2.2	A	5-2-40.3	A	5-2-62	A	5-2-89	A		
5-2-3	L	5-2-40.4	A	5-2-63	A	5-2-90.0	A		
5-2-4.1	A	5-2-40.5	A	5-2-64	A	5-2-90.1	A		
5-2-4.2	A	5-2-40.6	A	5-2-65	A	5-2-90.2	A		
5-2-5	K	5-2-40.7	A	5-2-66.1	A	5-2-91	K		
5-2-6.0	A	5-2-40.8	A	5-2-66.2	A	5-2-92.1	A		
5-2-6.1	A	5-2-41	L	5-2-66.3	A	5-2-92.2	A		
5-2-6.2	A	5-2-42.0	A	5-2-66.4	A	5-2-93	G		
5-2-7	K	5-2-42.1	A	5-2-66.5	A	5-2-94	J		
5-2-8.0	A	5-2-42.2	A	5-2-66.6	A	5-2-95	J		
5-2-8.1	A	5-2-43	B	5-2-66.7	A	Comment Sheet	L		
5-2-8.2	A	5-2-44.0	A	5-2-66.8	A	Back Cover	-		
5-2-9	K	5-2-44.1	A	5-2-67	K				
5-2-10	A	5-2-44.2	A	5-2-68.0	A				
5-2-11	K	5-2-44.3	A	5-2-68.1	A				
5-2-12.1	A	5-2-44.4	A	5-2-68.2	A				
5-2-12.2	A	5-2-44.5	A	5-2-68.3	A				
5-2-13	K	5-2-44.6	A	5-2-68.4	A				
5-2-14.1	A	5-2-45	B	5-2-68.5	A				
5-2-14.2	A	5-2-46.1	A	5-2-69	K				
5-2-15	B	5-2-46.2	A	5-2-71	A				
5-2-16.1	A	5-2-46.3	A	5-2-73	L				
5-2-16.2	A	5-2-46.4	A	5-2-75	K				
5-2-17	D	5-2-46.5	A	5-2-77	A				

PREFACE

This manual contains section 5, Theory and Diagrams, of the Hardware Maintenance Manual set that supports the CONTROL DATA® CYBER 170 Models 172, 173, 174 Central Processor Unit (CPU). The following equipments are covered:

AA107-A / AA131-B	Chassis 1, Model 172 Central Processor Unit
AT253-A	Central Processor Unit speed-up option (Model 172 to 173)
AD103-A/ AD105-B	CPU-1 Chassis (Model 174).

Section 5 is divided into two volumes containing three parts:

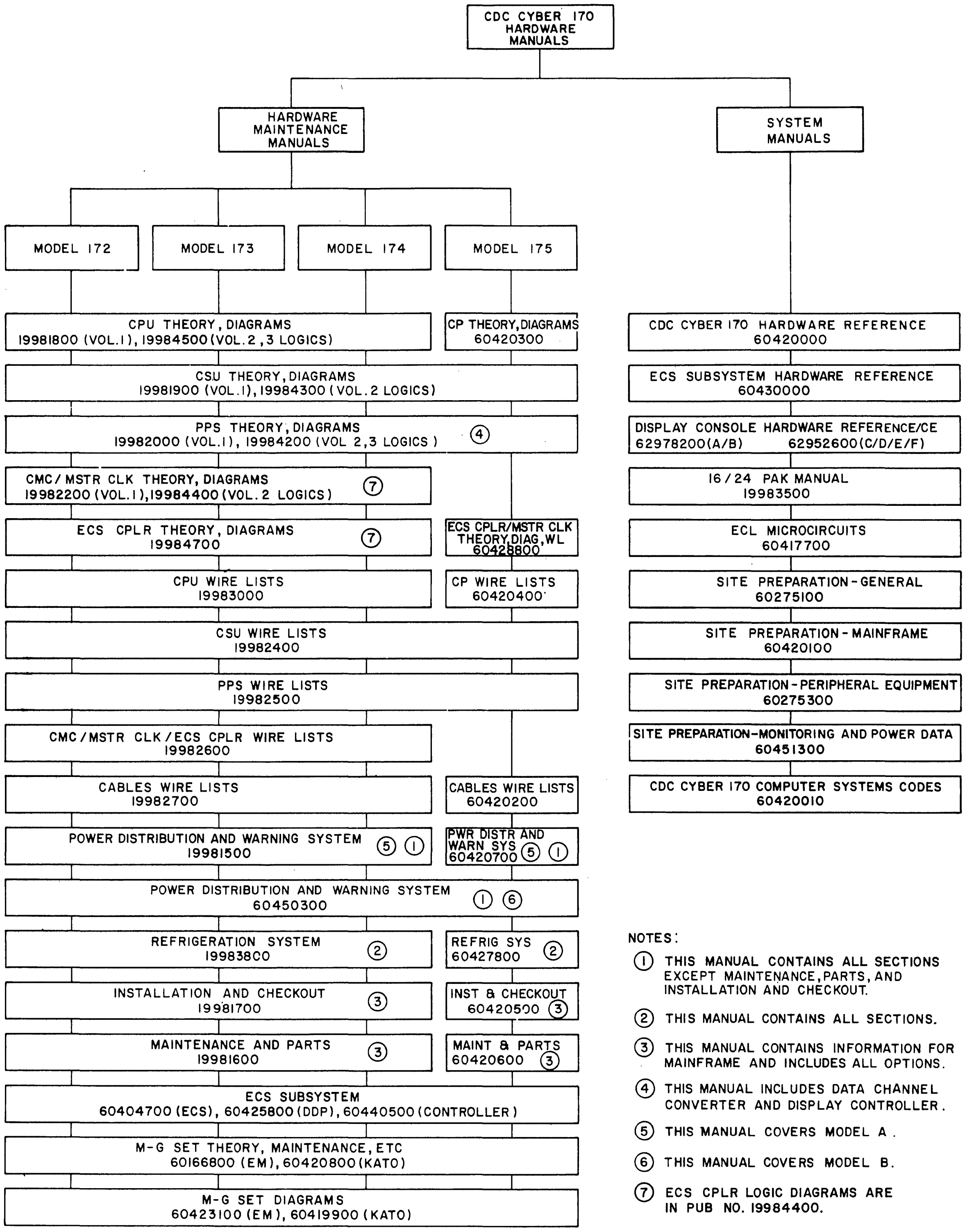
Volume 1	Part 1	Introduction
	Part 2	Theory of Operation and Block Diagrams
Volume 2	Part 3	Logic Diagram set.

The system publication index on the next page shows the relationship of this manual to others in the set. This index also graphically lists all other publications that support the CONTROL DATA CYBER 170 Mainframe Computer systems.

This manual also contains a manual to equipment correlation sheet that shows the equipment level (series code) covered by each manual revision.

NOTE:

This manual is revised only by ECOs affecting this manual. The correlation sheet, therefore, does not necessarily show the latest applicable equipment series code.


CDC CYBER 170 HARDWARE MANUALS
HARDWARE MAINTENANCE MANUALS
SYSTEM MANUALS
MODEL 172
MODEL 173
MODEL 174
MODEL 175
CPU THEORY, DIAGRAMS 19981800 (VOL.1), 19984500 (VOL.2, 3 LOGICS)
CP THEORY, DIAGRAMS 60420300
CSU THEORY, DIAGRAMS 19981900 (VOL.1), 19984300 (VOL.2 LOGICS)
PPS THEORY, DIAGRAMS 19982000 (VOL.1), 19984200 (VOL 2,3 LOGICS) ④
CMC/MSTR CLK THEORY, DIAGRAMS 19982200 (VOL.1), 19984400 (VOL.2 LOGICS) ⑦
ECS CPLR THEORY, DIAGRAMS 19984700 ⑦
ECS CPLR/MSTR CLK THEORY, DIAG, WL 60428800
CPU WIRE LISTS 19983000
CP WIRE LISTS 60420400
CSU WIRE LISTS 19982400
PPS WIRE LISTS 19982500
CMC/MSTR CLK/ECS CPLR WIRE LISTS 19982600
CABLES WIRE LISTS 19982700
CABLES WIRE LISTS 60420200
POWER DISTRIBUTION AND WARNING SYSTEM 19981500 ⑤ ①
PWR DISTR AND WARN SYS 60420700 ⑤ ①
POWER DISTRIBUTION AND WARNING SYSTEM 60450300 ① ⑥
REFRIGERATION SYSTEM 19983800 ②
REFRIG SYS 60427800 ②
INSTALLATION AND CHECKOUT 19981700 ③
INST & CHECKOUT 60420500 ③
MAINTENANCE AND PARTS 19981600 ③
MAINT & PARTS 60420600 ③
ECS SUBSYSTEM 60404700 (ECS), 60425800 (DDP), 60440500 (CONTROLLER)
M-G SET THEORY, MAINTENANCE, ETC 60166800 (EM), 60420800 (KATO)
M-G SET DIAGRAMS 60423100 (EM), 60419900 (KATO)
CDC CYBER 170 HARDWARE REFERENCE 60420000
ECS SUBSYSTEM HARDWARE REFERENCE 60430000
DISPLAY CONSOLE HARDWARE REFERENCE/CE 62978200(A/B) 62952600(C/D/E/F)
16/24 PAK MANUAL 19983500
ECL MICROCIRCUITS 60417700
SITE PREPARATION-GENERAL 60275100
SITE PREPARATION-MAINFRAME 60420100
SITE PREPARATION-PERIPHERAL EQUIPMENT 60275300
SITE PREPARATION-MONITORING AND POWER DATA 60451300
CDC CYBER 170 COMPUTER SYSTEMS CODES 60420010
1076

NOTES:

① THIS MANUAL CONTAINS ALL SECTIONS EXCEPT MAINTENANCE, PARTS, AND INSTALLATION AND CHECKOUT.

② THIS MANUAL CONTAINS ALL SECTIONS.

③ THIS MANUAL CONTAINS INFORMATION FOR MAINFRAME AND INCLUDES ALL OPTIONS.

④ THIS MANUAL INCLUDES DATA CHANNEL CONVERTER AND DISPLAY CONTROLLER.

⑤ THIS MANUAL COVERS MODEL A.

⑥ THIS MANUAL COVERS MODEL B.

⑦ ECS CPLR LOGIC DIAGRAMS ARE IN PUB NO. 19984400.

SYSTEM PUBLICATION INDEX

CONTENTS

FIGURES

SECTION 1

GENERAL DESCRIPTION

SECTION 2

OPERATION

SECTION 3

INSTALLATION AND CHECKOUT

Information for these sections is included in separate manuals. Refer to the system publication index at the front of this manual for publication numbers.

SECTION 4

THEORY OF OPERATION

(Included in Section 5)

SECTION 5

THEORY AND DIAGRAMS

SECTION 5

THEORY AND DIAGRAMS

Part 3: Logic Diagrams

(Included in Volumes 2 and 3)

SECTION 5

THEORY AND DIAGRAMS

Part 1: Introduction

CENTRAL PROCESSOR UNIT
INTRODUCTION

SYSTEM BLOCK DIAGRAM

Figure 5-1-1 shows the relationship of the CPU(s) to the rest of the CYBER 170 Models 172, 173, 174 mainframe computer systems.

MULTI-LEVEL THEORY AND DIAGRAMS

The theory and diagrams for the CPU are provided at three levels: primary block diagram; detailed pak diagram; logic diagram.

The primary block diagram shows a high-level relationship between the various functional components (registers, adders, etc.).

The next level (detailed pak diagram) is a functional-to-physical bridge that shows the data paths within the unit, a block diagram of each pak, all pak-interconnecting data and control signals, and test point charts for all data and address paths on the diagram.

The logic diagram set (most detailed level) depicts the unit at the integrated circuit (IC) level; it consists of one logic diagram for each pak in the unit.

Each logic diagram consists of IC identification and placement, IC interconnection, plus backpanel wiring information in the form of signal names and source/destination labels for each pin in the logic diagram.

PAK-TO-DIAGRAM CROSS REFERENCE TABLE

Table 5-1-1 lists all CPU pak types along with the following information on each:

Quantity	- number of paks of this type located in the CPU.
Location	- all chassis locations at which this pak type can be found.
Diagram	- detailed pak diagram(s) on which this pak type is shown.
Function	- the function covered on the corresponding pak diagram.

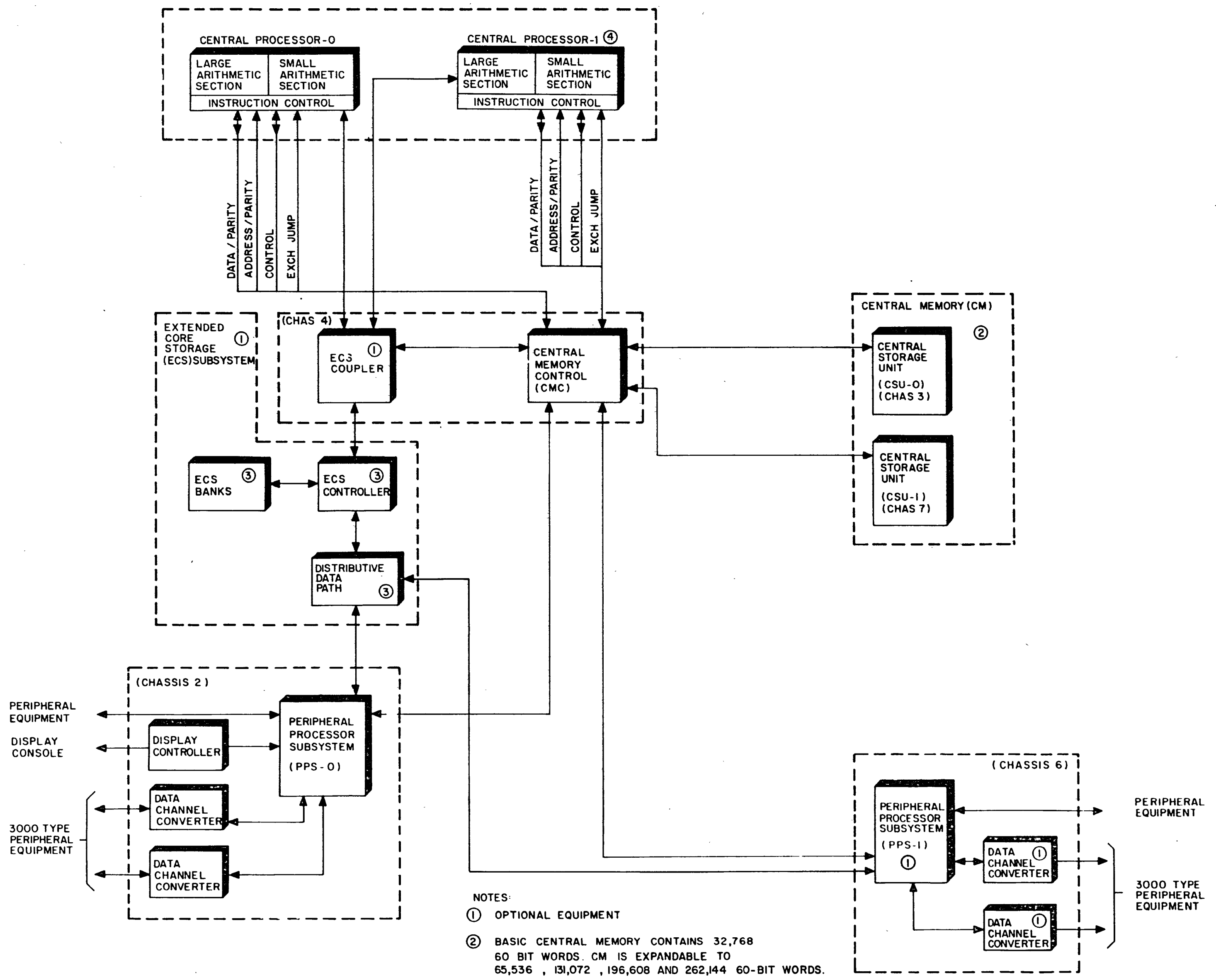
The table enables a user to identify all functional uses of a particular pak type, and shows the availability and location of substitute paks during maintenance; it also helps in locating all paks of the same type without having to scan the pak placement diagrams.

PAK PLACEMENT DIAGRAMS (figure 5-1-2)

These diagrams are included as an additional aid for translating the CPU into functional entities.

KEY TO BLOCK DIAGRAM SYMBOLS (figure 5-1-3)

Every effort has been made to keep the number of unfamiliar symbols on the block and detailed pak diagrams to a mimimum. The symbology and conventions have been chosen to simplify or clarify; they are therefore essential to the use and understanding of the diagrams. Note particularly that the AND and OR symbols define functions, not gates; e.g., AND gates in the hardware may actually be shown as OR functions on the diagrams.



NOTES:

① OPTIONAL EQUIPMENT

② BASIC CENTRAL MEMORY CONTAINS 32,768 60 BIT WORDS. CM IS EXPANDABLE TO 65,536, 131,072, 196,608 AND 262,144 60-BIT WORDS.

③ LOCATED EXTERNAL TO CENTRAL COMPUTER MAINFRAME

④ MODEL 174 ONLY

Figure 5-1-1. System Block Diagram

TABLE 5-1-1.1. PAK-TO-DIAGRAM CROSS REFERENCE

PAK TYPE	CHASSIS LOCATION	DPD REF.	FUNCTION
AA	C10	3.2	
	C18	3.4	
	I07, 18	3.5	
	F05, O26, I18,	3.8	
	F05, F11	3.13	
	M14	3.27	2ND STAGE CLOCK TUNING AND FNO
	M14	3.28	
	L12	3.29	
	N13	3.30	
	D35, E38, L41	3.45	
	C10, C18, D35, E25, E38, F05, F11, I07, I18, I27, K34, L12, L41, M14, N13, O25, K08	3.46	
AB	H27, H28, H29	3.46	PRIM CLK TUNING AND FNO
AD	L35	3.13	TIMING DELAY
	L35	3.14	RNI TIMING
	L35	3.17	WAIT II DLY TIMING
	L35	3.22	END CASE TIMING
	L35	3.24	FMD/FAD TIMING
	L35	3.27	ILL EXIT DLY TIMING
BZ	C21	3.1	U3 RGTR DISPLAY
CA	L36, L37, L38, L39, L40, M36, M37, M38, N36, N37, N38, O36, O37, O38, P36, P37, P38, Q36, Q37, Q38	3.0	
	B07, B18, C19, C20, C35	3.1	
	B05, B07, C05	3.2	
	F22, G22	3.3	FNO
	G26, H23, L18	3.4	
	G13, H22, H23	3.5	
	H30, J21, J22, K22	3.7	
	G05, H22, K22, K23	3.8	

TABLE 5-1-1.2. PAK-TO-DIAGRAM CROSS REFERENCE

PAK TYPE	CHASSIS LOCATION	DPD REF.	FUNCTION
CA	J37	3.9	
	N12	3.10	
	F36	3.12	
	F13, G11	3.13	
	I29, I32, J37	3.14	
	I29, L37, M33	3.16	
	F31, G36	3.17	
	C05, G35, G36, I24	3.18	
	C34, G22, H35, I32	3.19	
	C35, F22, F28, F30, F31 G22, G27, G29, G35	3.20	
	F31, G33, G35, G36	3.21	
	G33, H30, J22, J37	3.22	
	H22, H32, I24	3.23	
	H32, H35, I29, J30, J31, K31 K32	3.24	FNO
	H23, H35	3.25	
	G11, G13	3.26	
	H30, H32, N31	3.27	
	N31, M19, M20	3.28	
	L18, L26, M19, N14	3.29	
	N23, N24, M19, M20	3.30	
	M33, N11, N12, N14	3.31	
	M25, M33, N31	3.32	
	N29	3.34	
	L26, L36	3.35	
	M36, N29	3.36	
	L28	3.38	
	L28, N29, N31	3.40	
	L26	3.41	
	Q34	3.42	
	L26	3.43	
	M33	3.44	
	C34, C35, F31, F36, G27, G35	3.45	
EY	K08	3.8	CLOCK TUNING AND FNO

TABLE 5-1-1.3. PAK-TO-DIAGRAM CROSS REFERENCE

PAK TYPE	CHASSIS LOCATION	DPD REF.	FUNCTION
FA	H06, H11, H12, H15, I06, I12, I13, I19	3.6	CARRY LOOKAHEAD STAGE 1
	G12	3.13	CARRY LOOKAHEAD CKT XLTR
	N10	3.31	L ADDER LOOKAHEAD CKTS
FB	H13	3.6	L ADDER CARRY FUNCTION
	H13	3.13	F ADDER CARRY FUNCTION
FC	J02, J03, J04, J05, J06, J07 J08, J09, J10, J11, J12, J13 J14, J15, J16, J17, J18, J19, J20, K02, K03, K04, K05, K06, K07, K08, K09, K10, K11, K12, K13, K14, K15, K16, K17, K18, K19, K20, K21	3.8	I5 SELECTOR, C RGTR
FD	H02, H03, H04, H05, H06, H07 H08, H09, H10, H16, H17, H18, H19, I02, I03, I04, I05, I08, I09, I10, I11, I14, I15, I16, I17, I20, I21, I22	3.5	D ADDER LOGIC, D RGTR I4, I14 SELECTORS
FE	C11	3.2	EXPONENT TESTER
FF	F08, F09, F10, G08, G09	3.13	E, F RGTR, F ADDER, I2 SEL
FG	G16	3.9	FZ 128 TESTER
	G16	3.13	F DECODER
	G16	3.22	SHIFT TEST CKT
	G16	3.24	F TEST CKT
FH	P02, P03, P04, P05, P06, P07, P08, P09, P10, P11, P12, P13, P14, P15, P16, P17	3.10	SHIFT NETWORK RANK 2
	Q02, Q03, Q04, Q05, Q06, Q07, Q08, Q09, Q10, Q11, Q12, Q13, Q14, Q15, Q16, Q17	3.11	SHIFT NETWORK RANK 3
	Q15	3.13	
	R02, R03, R04, R05, R06, R07, R08, R09, R10, R11, R12, R13, R14, R15, R16	3.11	SHIFT NETWORK RANK 4
	P15	3.18	NZERO COMPLEMENTOR
	P04	3.19	MONFLG COMPLEMENTOR
	P11	3.25	BON2 COMPLEMENTOR
	Q09, Q10, Q11	3.46	TLKNN DLY CKT

TABLE 5-1-1.4. PAK-TO-DIAGRAM CROSS REFERENCE

PAK TYPE	CHASSIS LOCATION	DPD REF.	FUNCTION
FJ	O02, O03, O04, O05, O06, O07, O08, O09, O17, O18, O19, O20, O21, O22, O23, O24	3.10	SHIFT NETWORK RANK 1
FK	B19	3.1	FUNCTION DECODE CKT
FL	P22, P23, P24, P25, P26, P27, P28, P29	3.10	SHIFT NETWORK RANK 2
	Q22, Q23, Q24, Q25, Q26, Q27, Q28, Q29	3.11	SHIFT NETWORK RANK 3
	P30, R22, R23, R24, R25, R26, R27, R28, R29	3.11	SHIFT NETWORK RANK 4
FM	O10	3.9	FEQ 100 XLTR
	O10, O12, O13	3.10	NORM: ENCODER AND ZERO TEST CKT
FN	O11	3.10	NORMALIZE NETWORK
FP	D21, D22, D23, D24, D25, E21, E22, E23, E24	3.3	P, RA, MA, FL RGTR I0 SELECTOR
FQ	F19, F20, F21, G19, G20	3.4	RAE, FLE RGTR, I1 SELECTOR
FR	B02, B03, B04, B08, B09	3.2	A RGTR AND INPUT SELECTOR
	C02, C03, C04, C08, C09	3.2	B RGTR AND INPUT SELECTOR
	D02, D03, D04, D05, D09, D10, D11, E02, E03, E04, E05, E09, E10, E11, E12	3.2	X RGTR AND INPUT SELECTOR
FS	P18, P19	3.9	I9 SELECTOR AND SK COUNTER
FT	C27	3.17	AOR TEST CKT
FU	B13, B14, B15, B16, C13, C14, C15, C16	3.1	U1, RNI, U3 RGTR U2 SELECTOR
FV	J35	3.1	CONSTANT GENERATOR
	J35	3.14	PARCEL COUNT DECODER
	J35	3.24	SIGN TEST CKT
FW	C26	3.4	EE, EM RGTR AND SELECTOR

TABLE 5-1-1.5. PAK-TO-DIAGRAM CROSS REFERENCE

PAK TYPE	CHASSIS LOCATION	DPD REF.	FUNCTION
FX	K37	3.0	
	B20, C33	3.1	
	D07, D08, E06, E07	3.2	
	D26, D27	3.3	
	H20, H21, H24	3.5	
	J23, J24, J25	3.7	
	K24, K25	3.8	
	O15, P20, P21, Q19, Q20, Q21, R19, R20, R21	3.9	FNO AND COMPLEMENTORS
	O14	3.10	
	D16, D17, D18, E17, E18	3.12	
	F14, F15, F16, F17, F18, G14, G15, G16, G17	3.13	
	B17, C17, J34	3.14	
	G37, O16	3.15	
	B17	3.16	
	C17, P39	3.17	
	Q18, R19	3.20	
	H25, H26, I23, I26	3.23	
	L24, M07, M08, M19, M20	3.28	
	L23	3.29	
	L32, M32, N32	3.32	
	D33	3.46	
FY	E14, E15, E19, E20	3.12	I3 INPUT SEL, COMP CONTROL
FZ	D36, E31, E32, E33, E34, E35	3.45	I7XLTR, HR RGTR, PAR GEN
GA	G31	3.21	INCREMENT SEQ TIMING CHAIN
GB	G32	3.21	INCREMENT RGTR CONTROLS

TABLE 5-1-1.6. PAK-TO-DIAGRAM CROSS REFERENCE

PAK TYPE	CHASSIS LOCATION	DPD REF.	FUNCTION
GC	C06, E37	3.1	
	C06, E08, F12	3.2	
	E27	3.3	
	I25	3.5	
	I25	3.8	
	I37, J36	3.9	
	E13, E16, N33	3.12	
	E13, F12, G10, H41	3.13	
	I37, J36, I28	3.14	
	E37, J37	3.17	MISC XLTRS
	H41	3.19	
	I28, K36	3.25	
	L25	3.27	
	L25	3.28	
	L25	3.29	
	M35	3.22	
	N25	3.30	
	L25, M24	3.31	
	N33	3.35	
	N33	3.42	
	N33	3.44	
	N36, N37, E36	3.45	
	O31	3.14	
GD	G34	3.18	JUMP SEQ TIMING CHAIN
GE	F27	3.20	EXCH JUMP ENAB CKT
GF	F29, G28, G30	3.20	EXCH JUMP TIMING CHAIN
GG	I31	3.14	RNI START AND SEQ CONTROL
GH	I30	3.14	RNI CONTROL XLTR
GJ	I33	3.19	RET JUMP SEQ TIMING
GK	B06, C07, D06	3.2	A, B, X ADRS SELECTORS

TABLE 5-1-1.7. PAK-TO-DIAGRAM CROSS REFERENCE

PAK TYPE	CHASSIS LOCATION	DPD REF.	FUNCTION
GL	I35	3.14	PARCEL COUNTER
	I35	3.17	AOR SEQ TIMING CHAIN
GM	I34	3.16	ACCEPT SEQ TIMING CHAIN
GN	J32	3.24	FMD TIMING & EXIT DECODER
GP	J28	3.25	ENABLE I5 XLTR
	J28	3.26	ENABLE I3, I2 AND SK CONTROLS
GQ	J29	3.23	ALU FUNCTION ENABLE CKT
	J29	3.25	ENABLE CLOCK C XLTR
	J29	3.26	F ALU FUNCTION CODE SEL
GR	K29	3.26	F ALU DECODER ENAB CKTS
GS	K30	3.25	I5 COMP CONTROL XLTR
	K30	3.26	I3, I5 RGTR ENAB CKTS
GT	H38	3.15	COMM TIME SEQUENCE CKT
	H38	3.24	FUNCTION DECODER AND CONTROLS
GU	H37	3.15	COMM TIME FUNCTION DECODER
GV	H39	3.15	COMM TIME FUNCTION DECODER
	H39	3.22	RGTR ENAB CKT (SHIFT SEQ)
GW	H34	3.22	SHIFT SEQ TIMING CHAIN
GX	J27	3.5	I4 CONTROL SELECTOR
	J26, J27, K26, K27	3.7	I5 CONTROL XLTR
	J26, K28	3.8	I5 CONTROL XLTR
	K28	3.19	SETILN XLTR
	K26	3.22	COEFQ0 XLTR
	K27	3.24	EXSR2 XLTR

TABLE 5-1-1.8. PAK-TO-DIAGRAM CROSS REFERENCE

PAK TYPE	CHASSIS LOCATION	DPD REF.	FUNCTION
GY	H33	3.22	I3, I5 COMP XLTRS
	H33	3.23	BOOLEAN SEQ TIMING CHAIN
GZ	H31	3.27	ECS TIMING CHAIN
HA	J33	3.25	LOAD C RGTR, I5 XLTR
HB	F34	3.17	
HC	H14	3.6	CARRY LOOKAHEAD CKT (STAGE 2B)
HD	K35	3.14	SEQ EXIT DLY
HF	H36	3.15	FNO (COMT50)
	I36	3.9	SLI91 XLTR
	J21	3.7	FNO (I5S1)
	J22	3.7, 3.22	FNO (I5S2, SI52)
	K22	3.8	FNO (I5C67)
	K23	3.8	FNO (I5C85)
	E28	3.20	FNO (NEX 14, NCR9AB, ABI7)
HP	K35	3.14	SPEED UP MODULE
HN	K33	3.24	FAD TIMING AND EXIT DECODER
HQ	O30	3.32	CMU ON FF
	O30	3.33	ADRS SEQ, K1 ADRS FF
HS	G38	3.10, 3.25 3.26	
HT	M32	3.39	DATA COUNTER C/M SEQ
HU	M31	3.38	C/M DATA SEQ ENAB XLTR
HV	M28	3.40	C/M DATA SEQ COMMAND XLTR
HW	M29	3.40	C/M DATA SEQ ENAB XLTR
HX	L34	3.44	C/M EXIT SEQ CONTROL XLTR
HY	L31	3.42	COMPARE SEQ CONTROL XLTR
	L31	3.43	COLLATE TIMING SEQ
HZ	L27	3.43	COLLATE SEQ TIMING CHAIN

TABLE 5-1-1.9 PAK-TO-DIAGRAM CROSS REFERENCE

PAK TYPE	CHASSIS LOCATION	DPD REF.	FUNCTION
JA	G02, G03, G04, G06, G07, F02, F03, F04, F06, F07	3.8	H RGTR, I5 SELECTOR
JB	L02, L03, L04, L05, L06, L07, L08, L09, L10, L11	3.28	S RGTR, I31 SEL, COMPARATOR
JL	M03, M04, M05, M06, M09, M10, M11, M12, M13	3.28	R, Q RGTRS, I30 SELECTOR
JD	M16, M17, M18, M21, M22, M23	3.29	TS, TQ RGTRS, I32, 33, 37 SELECTORS
JE	L22	3.29	WP RGTR, I35 SEL, PRIORITY SEL
JF	L14	3.28	CP RGTR
JG	L16, L17, L18, L19, L20, L21	3.29	T RGTR, I34 SELECTOR
JH	L33	3.32	C/M INSTRUCTION DECODE ENAB
JJ	L13	3.28	FORCE EQUIVALENCE DECODER
	N19	3.30	I40 PRIORITY DECODER
JK	F23, F24, F25, G23, G24, G25	3.3	K1, K2 RGTRS, I49 SELECTOR
	F23, F24, F25, G23, G24, G25	3.9	I19 SELECTOR
JL	M26	3.32	C/M START SEQ INSTRUCTION DECODER
JM	N06, N07, N08, N09	3.31	LE, LF RGTRS, I46 SELECTOR
JN	N02, N03, N04, N05	3.31	LA, LC RGTRS, I45 SELECTOR
JP	N30	3.33	C/M ADRS SEQ TIMING CHAIN
JQ	N26	3.36	C/M ENABLE XLTR
JR	N27	3.34	C/M ADRS SEQ TIMING CHAIN
JS	N28	3.35	C/M ADRS FFs

TABLE 5-1-1..10 PAK-TO-DIAGRAM CROSS REFERENCE

PAK TYPE	CHASSIS LOCATION	DPD REF.	FUNCTION
JT	L29	3.41	C/M SHORT DATA SEQ TIMER
JU	L30	3.41	C/M SHORT DATA SEQ CONTROLS
JV	M27	3.37	C/M BUFFER CNTR MODE XLTR
JW	M30	3.38	C/M DATA SEQ TIMING CHAIN
JX	N15, N16, N17, N18	3.30	I40, 41, 42, 44 SELECTORS
JY	M15	3.30	C ALU AND SCR XLTR
JZ	N20, N21, N22	3.30	CSR, PW RGTRS, I47 SELECTOR
KC	N39	3.0	INPUT DATA PARITY CHECKER
	D37, F39, F40, I38, J38	3.45	OUTPUT DATA PARITY GEN
KR	M39, M40, M41, N40, N41 O39, O40, O41, G40, G41, P40	3.0	RCVRS, CR9 RGTR
KT	B39, B40, B41, C39, C40, C41, D39, D40, D41, E39, E40, E41, F41, I39, I40, I41, J39, J40, J41, K39, K40, K41	3.45	OUTPUT RGTR AND XMTRS
XA	H42	3.46	CLOCK TUNING AND FANOUT
XB	H27, 28, 29	3.46	CLOCK TUNING AND FANOUT

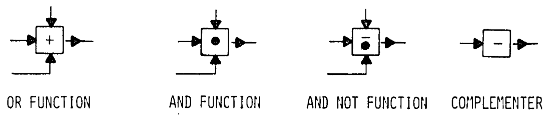
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	
A																						A
B		FRO	FRO	FRO	CAO	GKO	CAO	FRO	FRO				FUO	FUO	FUO	FUO	FXO	CAO	FKO	FXO		B
C		FRO	FRO	FRO	CAO	GCO	GKO	FRO	FRO	AAO	FEO		FUO	FUO	FUO	FUO	FXO	AAO	CAO	CAO	BZO	C
D		FRO	FRO	FRO	FRO	GKO	FXO	FXO	FRO	FRO	FRO	CPO	FYO	FYO	FYO	FXO	FXO	FXO	FYO	FYO	FPO	D
E		FRO	FRO	FRO	FRO	FXO	FXO	GCO	FRO	FRO	FRO	FRO	GCO	FYO	FYO	GCO	FXO	FXO	FYO	FYO	FPO	E
F		JAO	JAO	JAO	AAO	JAO	JAO	FFO	FFO	FFO	AAO	GCO	CAO	FXO	FXO	FXO	FXO	FXO	FQO	FQO	FQO	F
G		JAO	JAO	JAO	CAO	JAO	JAO	FFO	FFO	GCO	CAO	FAO	CAO	FXO	FXO	FGO	FXO	FXO	FQO	FQO		G
H		FDO	FDO	FDO	FDO	FAO	FDO	FDO	FDO	FDO	FAO	FAO	FBO	HCO	FAO	FDO	FDO	FDO	FDO	FXO	FXO	H
I		FDO	FDO	FDO	FDO	FAO	AAO	FDO	FDO	FDO	FDO	FAO	FAO	FDO	FDO	FDO	FDO	AAO	FAO	FDO	FDO	I
J		FCO	FCO	FCO	FCO	FCO	FCO	FCO	FCO	FCO	FCO	FCO	FCO	FCO	FCO	FCO	FCO	FCO	FCO	FCO	HFO	J
K		FCO	FCO	FCO	FCO	FCO	FCO	EYO	FCO	FCO	FCO	FCO	FCO	FCO	FCO	FCO	FCO	FCO	FCO	FCO	FCO	K
L		JBO	JBO	JBO	JBO	JBO	JBO	JBO	JBO	JBO	JBO	AAO	JJO	JFO	JGO	JGO	JGO	CAO	JGO	JGO	JGO	L
M		JCO	JCO	JCO	JCO	JCO	FXO	FXO	JCO	JCO	JCO	JCO	JCO	AAO	JYO	JDO	JDO	JDO	CAO	CAO	JDO	M
N		JNO	JNO	JNO	JNO	JMO	JMO	JMO	JMO	FAO	CAO	CAO	AAO	CAO	JXO	JXO	JXO	JXO	JJO	JZO	JZO	N
O		FJO	FJO	FJO	FJO	FJO	FJO	FJO	FJO	FMO	FNO	FMO	FMO	FXO	FXO	FXO	FJO	FJO	FJO	FJO	FJO	O
P		FHO	FHO	FHO	FHO	FHO	FHO	FHO	FHO	FHO	FHO	FHO	FHO	FHO	FHO	FHO	FHO	FSO	FSO	FXO	FXO	P
Q		FHO	FHO	FHO	FHO	FHO	FHO	FHO	FHO	FHO	FHO	FHO	FHO	FHO	FHO	FHO	FHO	FXO	FXO	FXO	FXO	Q
R		FHO	FHO	FHO	FHO	FHO	FHO	FHO	FHO	FHO	FHO	FHO	FHO	FHO	FHO	FHO		FXO	FXO	FXO	FXO	R
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	

Figure 5-1-2.1. Pak Placement Diagram - CPU

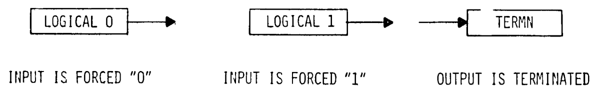
	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	
A																						A
B										FZO	FZO	FZO	FZO	FZO	FZO	FZO		KTO	KTO	KTO	PLG △2	B
C					FWO	FTO			FZO	FZO	FZO	FXO	CAO	CAO	FZO			KTO	KTO	KTO		C
D	FPO	FPO	FPO	FPO	FXO	FXO	FXO			FZO	FZO	FXO	FZO	AAO	FZO	KCO		KTO	KTO	KTO		D
E	FPO	FPO	FPO	AAO	FXO	GCO	HFO			FZO	FZO	FZO	FZO	FZO	GCO	GCO	AAO	KTO	KTO	KTO		E
F	CAO	JKO	JKO	JKO	CPO	GEO	CAO	GFO	CAO	CAO			HBO		CAO			KCO	KCO	KTO		F
G	CAO	JKO	JKO	JKO	CAO	CAO	GFO	CAO	GFO	GAO	GBO	CAO	GDO	CAO	CAO	FXO	HSO		KRO	KRO		G
H	CAO	CAO	HFO	FXO	FXO	ABO △1XBO	ABO △1XBO	ABO △1XBO	CAO	GZO	HFO	GYO	GWO	CAO	HFO	GUO	GTO	GVO	KTO	GCO	XAO △1	H
I	FDO	FXO	CAO	GCO	FXO	AAO	GCO	CAO	GHO	GGO	CAO	GJO	GMO	GLO	HFO	GCO	KCO	KTO	KTO	KTO		I
J	HFO	FXO	FXO	FXO	GXO	GXO	GPO	GQO	CAO	CAO	GNO	HAO	FXO	FVO	GCO	CAO	LDO	KTO	KTO	KTO		J
K	HFO	HFO	FXO	FXO	GXO	GXO	GXO	GRO	GSO	CAO	CAO	HNO	AAO	HDO/ HPO	GCO	FXO	BZO	EZO	EZO	EZO		K
L	JEO	FXO	FXO	GCO	CAO	HZO	CAO	JTO	JUO	HYO	FXO	JHO	HXO	ADO	CAO	CAO	CAO	CAO	CAO	AAO		L
M	JDO	JDO	GCO	CAO	JLO	JVO	HVO	HWO	JWO	HUO	HTO	CAO	FXO	GCO	CAO	CAO	CAO	KRO	KRO	KRO		M
N	JZO	CAO	CAO	GCO	JQO	JRO	JSO	CAO	JPO	CAO	FXO	GCO	HTO		CAO	CAO	CAO	KCO	KRO	KRO		N
O	FJO	FJO	FJO	AAO				AEO	HQO	GCO			CAO		CAO	CAO	CAO	KRO	KRO	KRO		O
P	FLO	FLO	FLO	FLO	FLO	FLO	FLO	FLO	FLO						CAO	CAO	CAO	FXO	KRO			P
Q	FLO	FLO	FLO	FLO	FLO	FLO	FLO	FLO							CAO	CAO	CAO					Q
R	FLO	FLO	FLO	FLO	FLO	FLO	FLO	FLO										△1 MODEL B ONLY· △2 HEAT SENSING MODULE MODEL B ONLY·				R
	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	

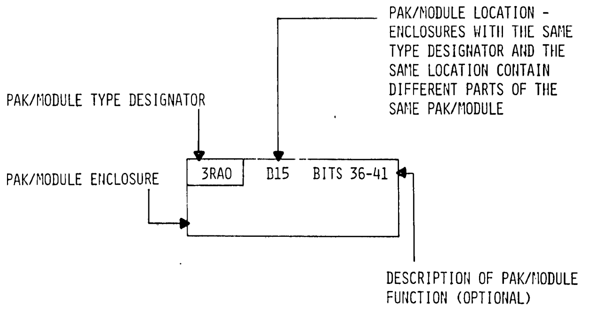
Figure 5-1-2.2. Pak Placement Diagram - CPU

SYMBOLS



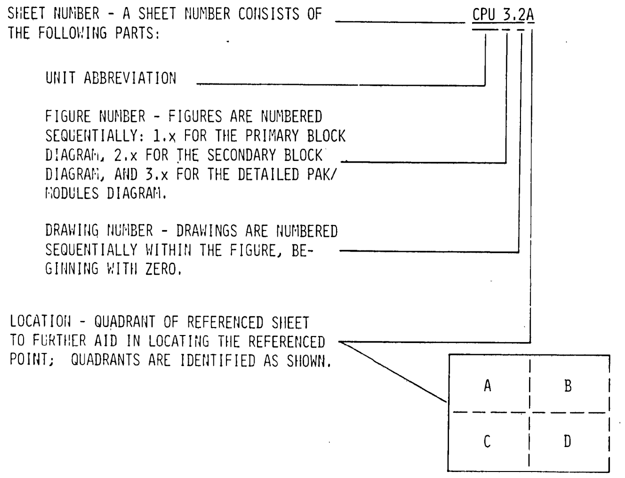
THESE SYMBOLS DENOTE THE FUNCTION BEING PERFORMED REGARDLESS OF THE TYPE OF GATE BEING USED IN THE LOGIC CIRCUIT. THUS, THE AND FUNCTION IS ENABLED BY COINCIDENT INPUTS AND THE OR FUNCTION IS SATISFIED BY THE PRESENCE OF EITHER INPUT.




PAK/MODULE LOCATION - ENCLOSURES WITH THE SAME TYPE DESIGNATOR AND THE SAME LOCATION CONTAIN DIFFERENT PARTS OF THE SAME PAK/MODULE
PAK/MODULE TYPE DESIGNATOR
PAK/MODULE ENCLOSURE
3RAO
D15
BITS 36-41
DESCRIPTION OF PAK/MODULE FUNCTION (OPTIONAL)

REFERENCES

A REFERENCE IS LOCATED AT THE BEGINNING OR END OF ANY LEAD THAT HAS ITS ORIGIN OR DESTINATION ON A PAK/MODULE NOT READILY ACCESSIBLE TO THE LEAD. MOST REFERENCES ARE MADE TO OTHER SHEETS WITHIN THE UNIT OR TO SHEETS IN OTHER UNITS, BUT A REFERENCE MAY BE TO ANOTHER POINT ON THE SAME SHEET. A REFERENCE CONSISTS OF THE FOLLOWING INFORMATION:



SHEET NUMBER - A SHEET NUMBER CONSISTS OF THE FOLLOWING PARTS: CPU 3.2A

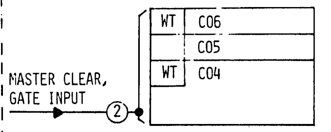
UNIT ABBREVIATION

FIGURE NUMBER - FIGURES ARE NUMBERED SEQUENTIALLY: 1.x FOR THE PRIMARY BLOCK DIAGRAM, 2.x FOR THE SECONDARY BLOCK DIAGRAM, AND 3.x FOR THE DETAILED PAK/MODULES DIAGRAM.

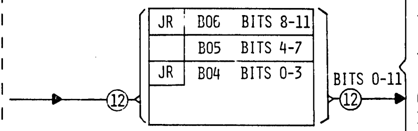
DRAWING NUMBER - DRAWINGS ARE NUMBERED SEQUENTIALLY WITHIN THE FIGURE, BEGINNING WITH ZERO.

LOCATION - QUADRANT OF REFERENCED SHEET TO FURTHER AID IN LOCATING THE REFERENCED POINT; QUADRANTS ARE IDENTIFIED AS SHOWN.

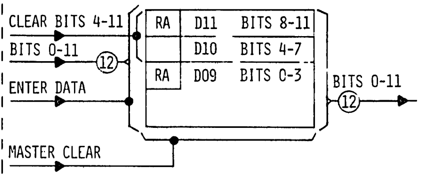
STACKED MODULES I/O CONVENTIONS



A SOLDER-POINT CONNECTION TO A BRACKET DENOTES THAT ALL OF THE BITS OR SIGNALS ON THE LEAD ARE INPUTS TO ALL PAKS/MODULES ADJACENT TO THE BRACKET; I.E., PARALLEL INPUTS



THIS TYPE OF CONNECTION ON AN INPUT BRACKET DENOTES THAT THE BITS OR SIGNALS ON THE LEAD ARE DISTRIBUTED AMONG THE PAKS/MODULES ADJACENT TO THE BRACKET. THE SPECIFIC INPUTS TO EACH PAK/MODULE IN THE STACK ARE IDENTIFIED WITHIN THE PAK/MODULE. THIS TYPE OF CONNECTION ON AN OUTPUT BRACKET DENOTES THAT ALL THE OUTPUTS FROM THE INCLUDED PAKS/MODULES MERGE INTO ONE LEAD. THE MERGING BITS OR SIGNALS ARE IDENTIFIED BY THE SIGNAL NAME ON THE LEAD.



ANY NUMBER OF CONNECTIONS OF EITHER TYPE MAY BE MADE TO THE SAME BRACKET.

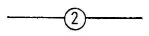
BRACKETS ARE NESTED WHEN CONNECTIONS TO ALL PAKS/MODULES IN THE STACK ARE NOT IDENTICAL.

BRACKETS ALONG TOP OR BOTTOM OF STACK ARE EQUIVALENT TO BRACKETS ADJACENT TO ALL PAKS/MODULES IN STACK.

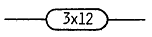
INDIVIDUAL INPUTS OR OUTPUTS (NOT BRACKETED) MAY ENTER OR LEAVE INDIVIDUAL PAKS/MODULES IN STACK THROUGH BRACKET.

SIGNAL BUNDLING CONVENTIONS

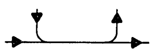
SIGNALS MAY BE COMBINED IN "HIGHWAYS"; THE NUMBER OF UNIQUE SIGNALS (NOT WIRES) IN THE "HIGHWAY" IS SHOWN IN A BALLOON :



MULTIPLE SETS OF SIGNALS IN A "HIGHWAY" ARE SHOWN IN AN OBLONG; E.G., 3 SETS OF 12 :



ENTRY OR EXIT OF A SIGNAL TO/FROM A HIGHWAY IS SHOWN WITH :



DISTRIBUTION (TAPPING) OF A SIGNAL FROM A HIGHWAY IS SHOWN WITH :

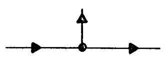


Figure 5-1-3. Key to Block Diagram Symbols

SECTION 5

THEORY AND DIAGRAMS

Part 2: Theory and Block Diagrams

INPUT REGISTER

The CR9 register receives all input data in the form of instructions or operands.

INSTRUCTION CONTROL REGISTERS

The U1, RNI, U2 and U3 registers form the portion of the processor that handles instructions. All instructions from CM are sent to CR9. From CR9, they continue to U1 where they are disassembled. From U1, instructions are sent to the RNI register, or via U2 to the U3 register. At U3, instructions are decoded and the appropriate control sequences are enabled to execute the instruction.

OPERATING AND CONTROL REGISTERS

The control registers include:

Program Address register	(P)
Reference Address register	(RA)
Monitor Address register	(MA)
Field Length register	(FL)
Reference Address for ECS register	(RAE)
Field Length for ECS register	(FLE)
Source Field Address register for compare/move	(K1)
Destination Field Address register for compare/move	(K2)
Exit Mode register	(EM)

These registers are used in conjunction with instruction controls during the execution of instructions.

The operating registers hold operands used during the execution of instructions. There are 24 operating registers divided into three groups of eight registers each: the address registers (A), the index registers (B), and the operand registers (X).

LARGE ARITHMETIC SECTION

The large arithmetic section consists of:

- 108-bit D adder
- 114-bit C register
- 108-bit D register
- Input selectors to the D register I14 and I4
- Input selectors to the C register I1, I15 and I5
- Shift network
- High/low select circuit
- Normalize network
- Input selectors to the iteration and shift counter I19 and I9
- SK shift and iteration counter.

The large arithmetic section is used during the execution of instructions using 60-bit operands. It includes all multiply, divide, logical, add, shift, compare/move, and ECS instructions.

SMALL ARITHMETIC SECTION

The small arithmetic section consists of:

- 18-bit F adder
- 18-bit E and F registers
- Input selectors to the F register I0, I3 and I2
- Input selectors to the E register I0, I3
- Address range test
- F register test circuits.

The small arithmetic section handles instructions using 18-bit operands; these include increment and jump instructions. This section also handles exponent manipulation for floating point instructions and compare/move address calculations.

COMPARE/MOVE DATA SECTION

The compare/move data section consists of:

- H, R, Q, S and T registers
- Unequal character position register (CP)
- Word comparison circuits
- Character comparison circuits
- Collate character comparison circuits
- TS, TQ and WP registers

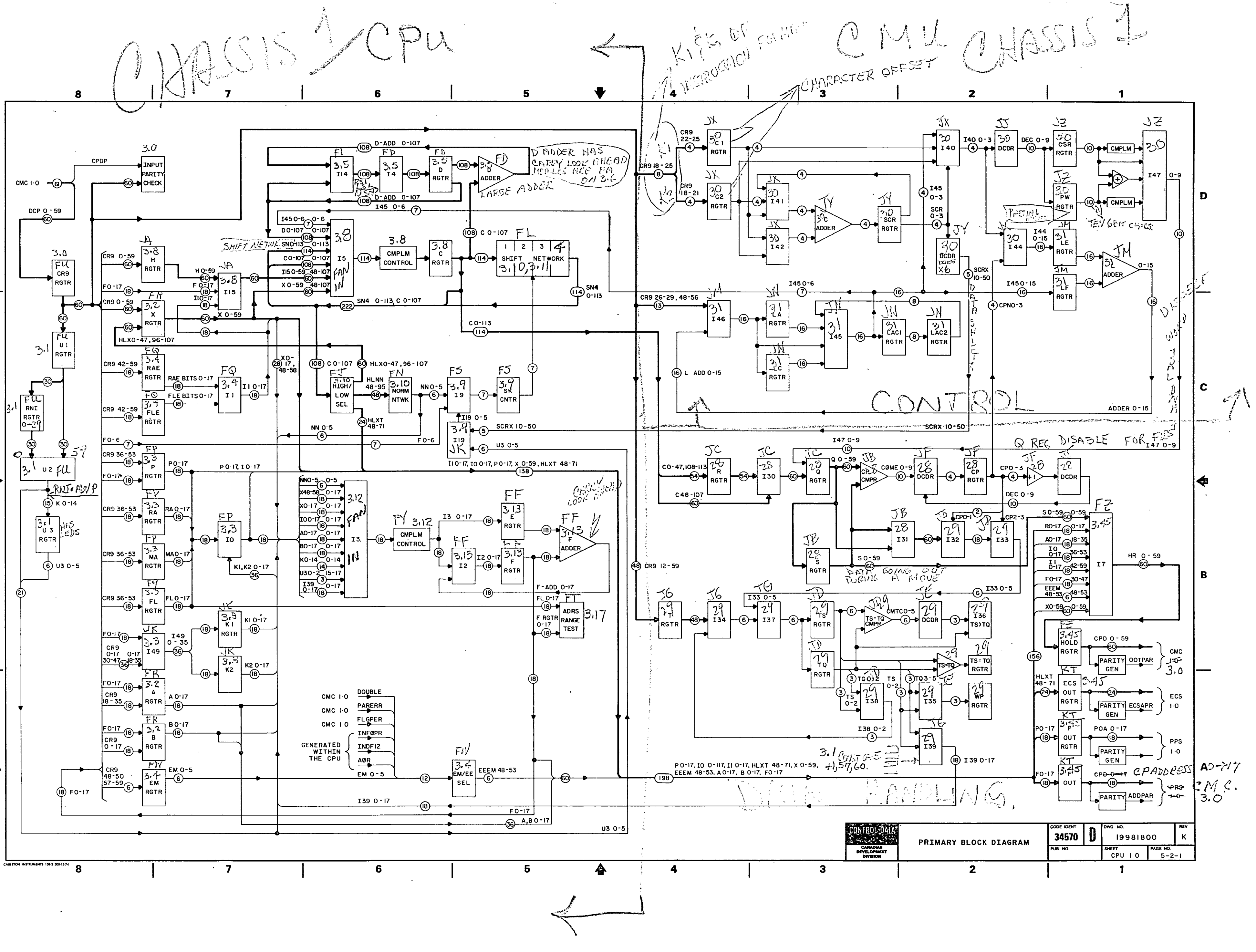
COMPARE/MOVE CONTROL SECTION

The compare/move control section consists of:

- C1 and C2 offset registers
- 4-bit C adder
- Shift count register (SCR)
- Character select (CSR) and partial write (PW) registers
- Field length registers LA, LC, LAC1 and LAC2
- L adder and LE and LF feeder registers.

OUTPUT SECTION

The processor output section consists of the hold register (HR), ECS output register, P output register and F output register. Data and control signals are sent from the output section to CMC, PPS and the ECS coupler.



CONTROL DATA
CANADIAN DEVELOPMENT DIVISION

PRIMARY BLOCK DIAGRAM

CODE IDENT		DWG NO.	REV
34570	D	19981800	K
PUB NO.		SHEET CPU 1 0	PAGE NO. 5-2-1

INPUT DATA AND CONTROL REGISTERS

The CR9 and input control registers located on the KR modules are the data input and control signal catching registers for the CPU.

CR9 REGISTER

The CR9 register receives 60-bit input operands or instructions from CMC. Parity for each 60-bit word is checked on the KC module with the input parity bit IDTPAR. Should an input parity error be detected, INPARE sets the input parity error FF during the accept sequence.

CONTROL REGISTERS

The input control registers receive control signals and parity information from CMC, PPS and the ECS Coupler. The control signals received are defined as follows:

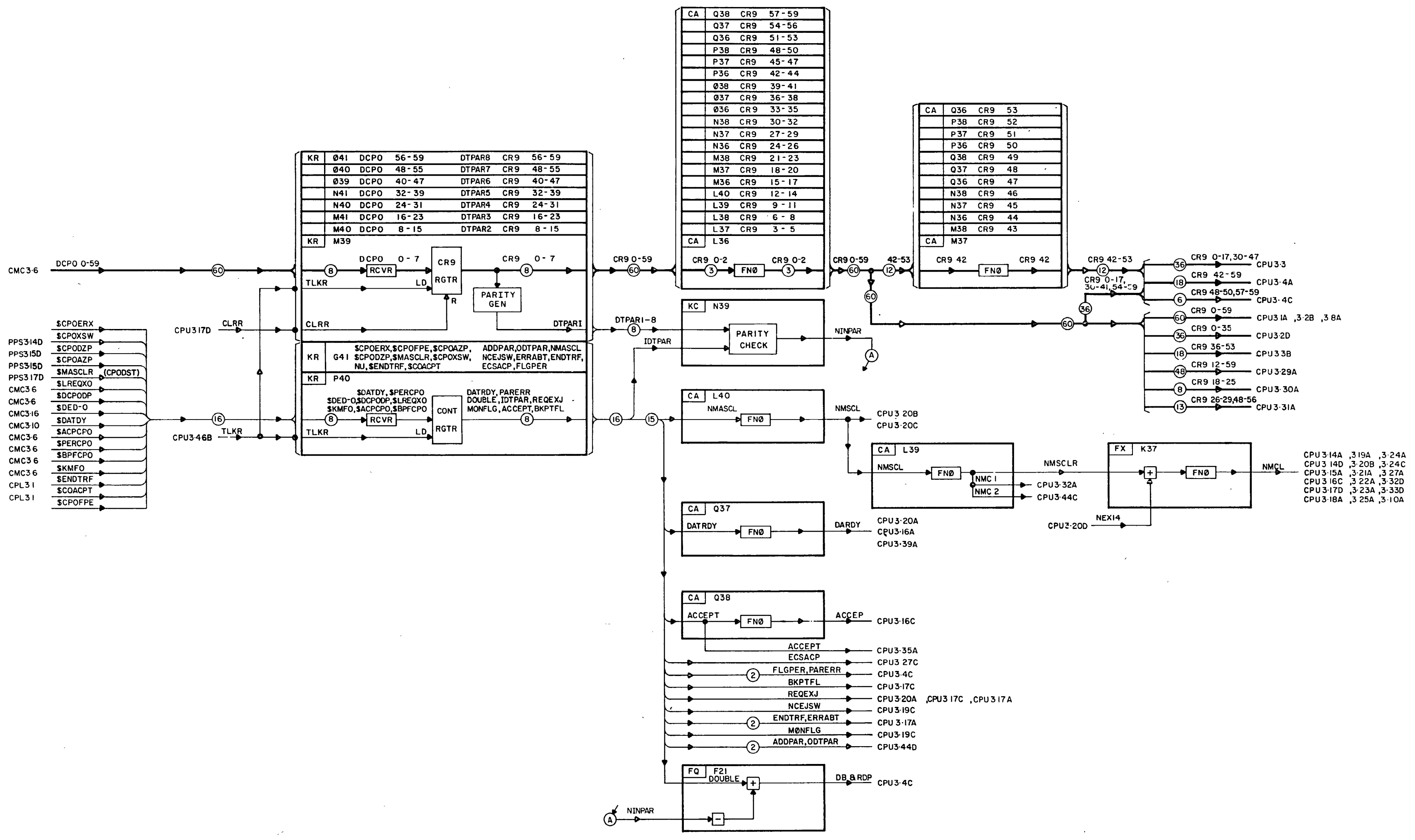
REQEXJ	Request Exchange: CMC requests the start of an exchange jump sequence.
IDTPAR	Input Data Parity: Data parity for 60 data bits sent to CR9.
DOUBLE	Double Data Error: Double data error detected by SECDED.
DATRDY	Data Ready: CMC sends data ready 50 ns ahead of output data.
ACCEPT	Accept: CMC sends accept when a request for CM access has been accepted and a memory cycle is initiated.
PARERR	Parity Error: A parity error detected in the input data or address information. This signal is sent from CMC at the same time as accept; however, a memory reference is inhibited.
BKPTFL	Breakpoint Flag: CMC sends the breakpoint flag when a breakpoint condition is met during a CPU to CM access.
MONFLG	Monitor Flag: This signal from CMC indicates the state of the monitor flag FF.
NCEJSW	MEJ /CEJ Switch: This signal from the PPS indicates whether the MEJ/CEJ switch from the dead start panel is in the ENABLE or DISABLE position.
NMASCL	Processor Master Clear
ENDTRF	ECS End Transfer: ECS coupler sends end transfer when an ECS transfer is completed normally.
ERRABT	ECS Error Abort End Transfer: ECS coupler sends error abort when an error condition has been detected.
ADDPAR	Zero Address Parity: This signal causes the parity bit, on addresses transmitted to the CMC, to be a constant zero.
OUTPAR	Zero Data Parity: This signal causes the parity bit, on data transmitted to the CMC, to be a constant zero.
FLGPER	Flag Register Operator Parity Error: Indicates detection by the CMC of a parity error on an ECS instruction affecting the flag register.
ECSACP	ECS Accept: Indicates readiness of the ECS coupler to process the ECS starting parameters from the CPU.

TABLE 5-2-1. CPU 3.0 KEY TEST POINTS

BIT NO	(CR9) KR			CA			CA		
	PAK LOC	IN	OUT	PAK LOC	IN	OUT	PAK LOC	IN	OUT
	1	DCP	$\overline{CR9}$		CR9	$\overline{CR9}$		CR9	$\overline{CR9}$
00	M39	03	01	L36	08	10,11			
01	M39	02		L36	03	2,1			
02	M39	05		L36	06	4			
03	M39	06		L37	08	10,11			
04	M39	13		L37	03	2,1			
05	M39	12	14	L37	06	4			
06	M39	09	10	L38	08	10,11			
07	M39	08	11	L38	03	2,1			
08	M40	03	01	L38	06	4			
09	M40	02		L39	08	10,11			
10	M40	05		L39	03	2,1			
11	M40	06		L39	06	4			
12	M40	13		L40	08	10,11			
13	M40	12	14	L40	03	2,1			
14	M40	09	10	L40	06	4			
15	M40	08	11	M36	08	10,11			
16	M41	03	01	M36	03	2,1			
17	M41	02		M36	06	4			
18	M41	05		M37	08	10,11			
19	M41	06		M37	03	2,1			
20	M41	13		06	4				
21	M41	12	14	M38	08	10,11			
22	M41	09	10	M38	03	2,1			
23	M41	08	11	M38	06	4			
24	N40	03	01	N36	08	10,11			
25	N40	02		N36	03	2,1			
26	N40	05		N36	06	4			
27	N40	06		N37	08	10,11			
28	N40	13		N37	03	2,1			
29	N40	12	14	N37	06	4			
30	N40	09	10	N38	08	10,11			
31	N40	08	11	N38	03	2,1			
32	N41	03	01	N38	06	4			
33	N41	02		O36	08	10,11			
34	N41	05		O36	03	2,1			
35	N41	06		O36	06	4			
36	N41	13		O37	08	10,11			
37	N41	12	14	O37	03	2,1			
38	N41	09	10	O37	06	4			
39	N41	08	11	O38	08	10,11			
40	O39	03	01	O38	03	2,1			
41	O39	02		O38	06	4			
42	O39	05		P36	08	10,11	M37	13	14
43	O39	06		P36	03	2,1	M38	13	14
44	O39	13		P36	06	4	N36	13	14
45	O39	12	14	P37	08	10,11	N37	13	14
46	O39	09	10	P37	03	2,1	N38	13	14
47	O39	08	11	P37	06	4	Q36	13	14
48	O40	03	01	P38	08	10,11	Q37	13	14
49	O40	02		P38	03	2,1	Q38	13	14
50	O40	05		P38	06	4	P36	13	14
51	O40	06		Q36	08	10,11	P37	13	14
52	O40	13		Q36	03	2,1	P38	13	14
53	O40	12	14	Q36	06	4	Q36	13	14
54	O40	09	10	Q37	08	10,11			
55	O40	08	11	Q37	03	2,1			
56	O41	03	01	Q37	06	4			
57	O41	02		Q38	08	10,11			
58	O41	05		Q38	03	2,1			
59	O41	06		Q38	06	4			

KR (CONTROL REGISTER)

SIGNAL	T.P.	PAK LOC	SIGNAL	T.P.	SIGNAL	T.P.	PAK LOC	SIGNAL	T.P.
DATDY	03	P40	$\overline{DATRDY}$	01	DED-O	08	P40		
ACPCPO	02	P40	$\overline{REQEXJ}$	14	ENDTRF	03	G41	$\overline{ENDTRF}$	01
PERCPO	05	P40	$\overline{IDTPAR}$	10	COACPT	05	G41	$\overline{NMASCL}$	14
BPFCPO	06	P40	DOUBLE	11	CPOXSW	13	G41	$\overline{ODTPAR}$	10
KMFO	13	P40			MASCLR	12	G41	$\overline{ADDPAR}$	11
LREQXO	12	P40			CPODZP	09	G41		
DCPODP	09	P40			CPOAZP	08	G41		



CONTROL DATA
CANADIAN DEVELOPMENT DIVISION

INPUT DATA AND CONTROL REGISTERS
CENTRAL PROCESSOR UNIT

CODE IDENT		DWG NO	REV
34570	D	19981800	L
PUB NO		SHEET CPU 3·0	PAGE NO 5-2-3

DETAILED PAK DIAGRAM (CPU 3.1)

U1, U2, RNI HOLD, U3, CONSTANT GENERATOR

The U1 register, RNI hold register, U2 selector and U3 register are contained on the FU module.

U1, RNI hold and U2 disassemble each 60-bit memory word into four 15-bit parcels. These parcels are then translated sequentially to determine the instruction format of the memory word.

U1 holds the 60-bit memory word from CR9 during RNI initial start or full RNI operations. Parcels 0 and 1 from U1 are gated sequentially through U2 to U3, and parcels 2 and 3 are stored in the RNI hold register during execution of a full RNI for the next four parcels from memory.

The output of U2 is sent to U3 for instruction translation. If during translation, parcel 0 in U3 is found to be part of a 30-bit instruction, parcel 1 will be gated into I3 directly from U2. This forms the K portion (K bits 0 - 14) of the instruction.

The U3 register feeds the function decode logic contained on the FK module, and provides instruction designator fanouts for the f, m, i, j and k bits of the instruction word.

An RNI exit to common time advances the parcel counter which gates the next parcel into U2. The next parcel RNI gates U2 to U3, translating the next instruction.

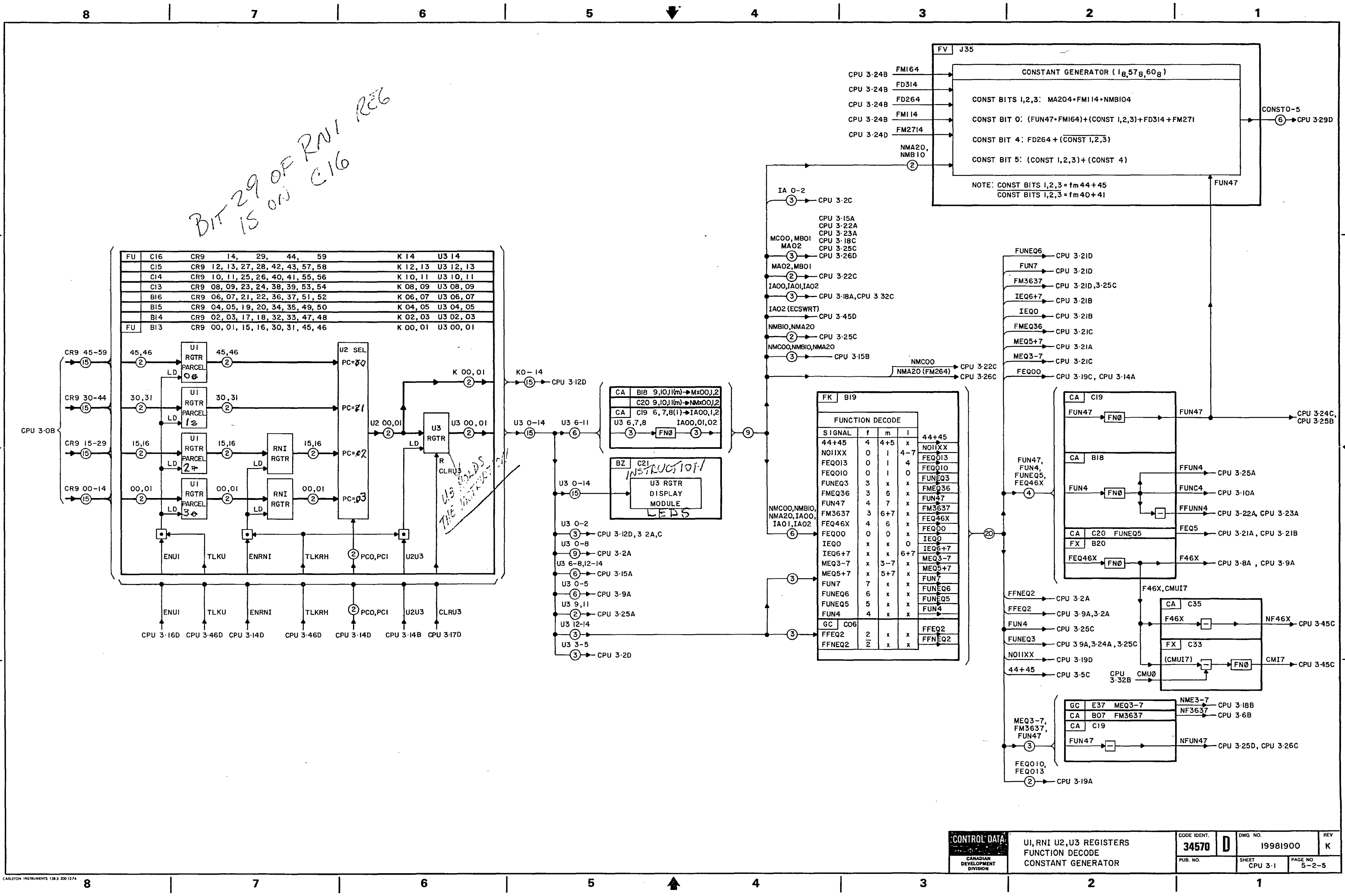
<u>CONSTANT GENERATOR</u>

The constant generator circuit located on the FV module generates constant values required by the FAD and FMD sequences. The constant generator is capable of generating constant values of 1, 57_8 and 60_8. The constant value is sent via I39 to I3.

TABLE 5-2-2. CPU 3.1 KEY TEST POINTS

	FU				CA		
BIT NO	PAK LOC	U1 OUT	RNI OUT	$\overline{K}$ OUT	PAK LOC	IN U3	OUT $\overline{IA/NM/M}$
00	B13	12	13	14			
01	B13	11	04	02			
02	B14	12	13	14			
03	B14	11	04	02			
04	B15	12	13	14			
05	B15	11	04	02			
06	B16	12	13	14	C19	08	10, 11
07	B16	11	04	02	C19	03	02, 01
08	C13	12	13	14	C19	06	04
09	C13	11	04	02	C20/B18	08	10, 11
10	C14	12	13	14	C20/B18	03	02, 01
11	C14	11	04	02	C20/B18	06	04
12	C15	12	13	14			
13	C15	11	04	02			
14	C16	12	13	14			
15	B13	07	09				
16	B13	06	01				
17	B14	07	09				
18	B14	06	01				
19	B15	07	09				
20	B15	06	01				
21	B16	07	09				
22	B16	06	01				
23	C13	07	09				
24	C13	06	01				
25	C14	07	09				
26	C14	06	01				
27	C15	07	09				
28	C15	06	01				
29	C16	07	09				
30	B13	10					
31	B13	03					
32	B14	10					
33	B14	03					
34	B15	10					
35	B15	03					
36	B16	10					
37	B16	03					
38	C13	10					
39	C13	03					
40	C14	10					
41	C14	03					
42	C15	10					
43	C15	03					
44	C16	10					
45	B13	08					
46	B13	05					
47	B14	08					
48	B14	05					
49	B15	08					
50	B15	05					
51	B16	08					
52	B16	05					
53	C13	08					
54	C13	05					
55	C14	08					
56	C14	05					
57	C15	08					
58	C15	05					
59	C16	08					

13-
00



BIT 29 OF RNI REG IS ON C16
INSTRUCTION
U3 RGTR DISPLAY MODULE
LEDS
U3 HOLDS THE INSTRUCTION
CONSTANT GENERATOR ($1_8, 57_8, 60_8$)
CONST BITS 1,2,3: MA204•FM114•NMB104
CONST BIT 0: (FUN47•FM164)+(CONST 1,2,3)+FD314+FM271
CONST BIT 4: FD264+(CONST 1,2,3)
CONST BIT 5: (CONST 1,2,3)+(CONST 4)
NOTE: CONST BITS 1,2,3 = fm44+45
CONST BITS 1,2,3 = fm40+41
CONSTO-5
U1 RGTR PARCEL
RNI RGTR
U2 SEL
U3 RGTR
ENU1
TLKU
ENRNI
TLKRH
PCO,PCI
U2U3
CLRU3
CPU 3·0B

FU	C16	CR9 14, 29, 44, 59	K 14	U3 14
	C15	CR9 12, 13, 27, 28, 42, 43, 57, 58	K 12, 13	U3 12, 13
	C14	CR9 10, 11, 25, 26, 40, 41, 55, 56	K 10, 11	U3 10, 11
	C13	CR9 08, 09, 23, 24, 38, 39, 53, 54	K 08, 09	U3 08, 09
	B16	CR9 06, 07, 21, 22, 36, 37, 51, 52	K 06, 07	U3 06, 07
	B15	CR9 04, 05, 19, 20, 34, 35, 49, 50	K 04, 05	U3 04, 05
	B14	CR9 02, 03, 17, 18, 32, 33, 47, 48	K 02, 03	U3 02, 03
FU	B13	CR9 00, 01, 15, 16, 30, 31, 45, 46	K 00, 01	U3 00, 01

FK B19

FUNCTION DECODE

SIGNAL	f	m	i
44+45	4	4+5	x
NO11XX	0	1	4-7
FEQ013	0	1	4
FEQ010	0	1	0
FUNEQ3	3	x	x
FMEQ36	3	6	x
FUN47	4	7	x
FM3637	3	6+7	x
FEQ46X	4	6	x
FEQ00	0	0	x
IEQ0	x	x	0
IEQ6+7	x	x	6+7
MEQ3-7	x	3-7	x
MEQ5+7	x	5+7	x
FUN7	7	x	x
FUNEQ6	6	x	x
FUNEQ5	5	x	x
FUN4	4	x	x
GC C06			
FFEQ2	2	x	x
FFNEQ2	$\overline{2}$	x	x

CONTROL DATA CANADIAN DEVELOPMENT DIVISION

UI, RNI U2, U3 REGISTERS
FUNCTION DECODE
CONSTANT GENERATOR

CODE IDENT.		DWG NO.	REV
34570	D	19981900	K
PUB. NO.		SHEET CPU 3·1	PAGE NO 5-2-5

CARLETON INSTRUMENTS 128-2 2001274

DETAILED PAK DIAGRAM (CPU 3.2)

X, A, B REGISTERS

X REGISTERS

The X registers provide buffering between memory and the execution hardware. All 60-bit operands needed for instruction execution are obtained from the X registers, and all 60-bit results are returned to the X registers.

A REGISTERS

The A registers provide means for moving data between memory and the X registers, and comprise eight 18-bit address registers (A0 - A7). An address placed in an address register (A1 - A5) causes an immediate central memory reference to that address, and loads the operand from memory in the corresponding X register (X1 - X5). Placing an address in one of the two remaining address registers (A6 - A7) stores the word from the corresponding X6 or X7 register in the central memory address specified by A6 or A7.

The A0 register operates independently of X0 in that a change to the contents of A0 does not initiate a central memory reference.

B REGISTERS

The B registers consist of eight 18-bit indexing registers that have no connection to central memory. The B registers are manipulated by increment, pack and unpack instructions and can be used for control of program branching. B0 is maintained as a constant zero index.

X, A, B LOGIC LAYOUT

The X, A and B registers are contained on the FR modules. Each module consists of 4 ~~MOS~~ RAM memory chips providing 16x4 bits of register storage. Only 8x4 bits are used on each module.

C, A and B register input selection is provided on each FR module. The X registers can receive input data from CR9 during exchange jump operations, or from the high/low C register select circuit. The A and B registers can receive input data from CR9 during exchange jump operations, or from the F register. The TLK (write) strobe is used to load the register, preventing a continuous read of the selected register.

X, A, B SELECTION

To read or write the contents of any one of the eight X, A or B registers, a 3-bit address is generated from the GK module. The GK module allows selection of the i, j or k portions of the instruction in U3 to be used for register addressing. For exchange jump operations, the GK module provides a sequence decode circuit to generate the required register addresses.

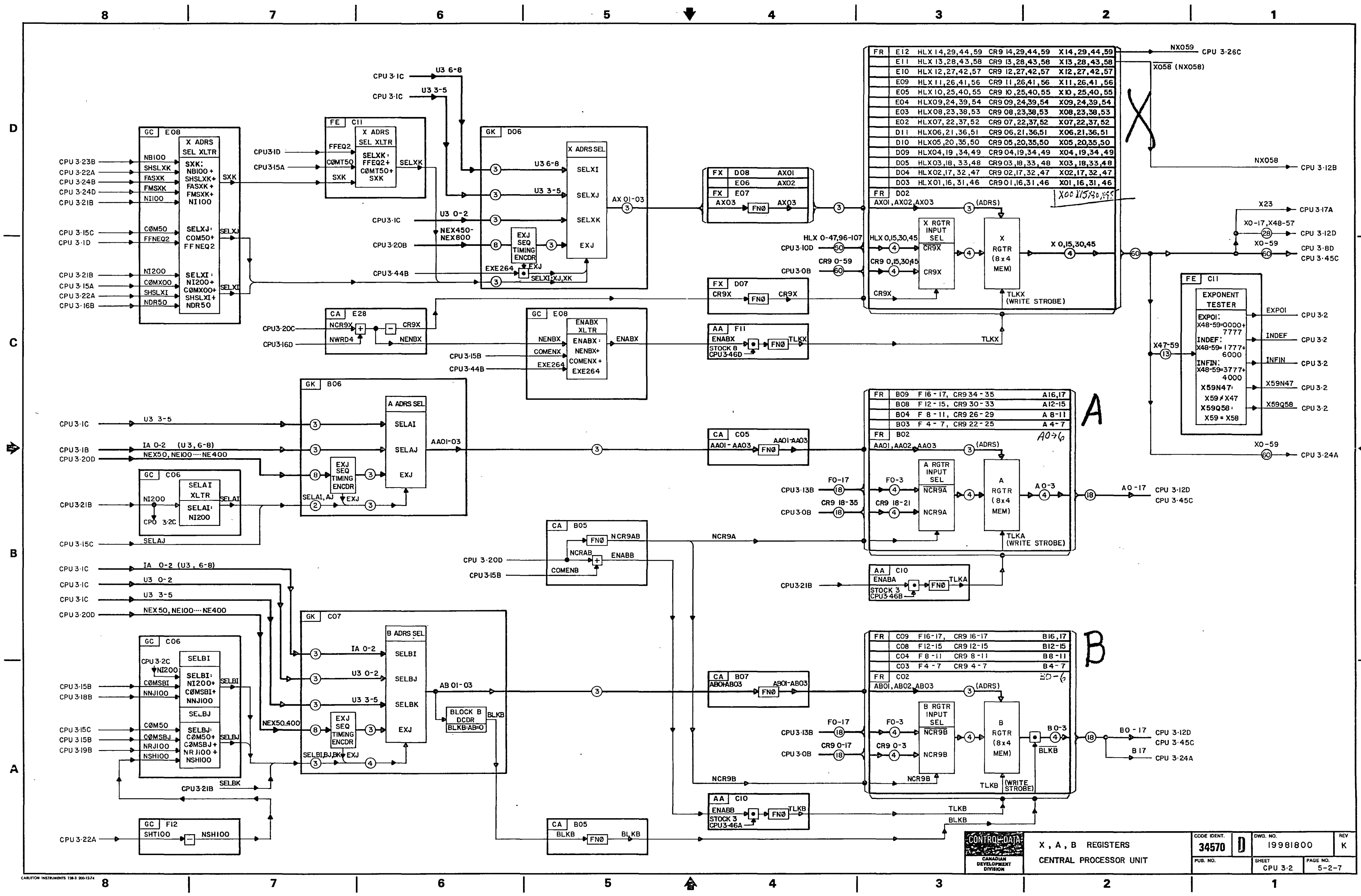
TABLE 5-2-3. CPU 3.2 KEY TEST POINTS

	GK		FX			FR (X REGISTER)		
BIT NO	PAK LOC	OUTPUT $\overline{AX}$	PAK LOC	INPUT AX	OUTPUT $\overline{AX}$	PAK LOC	X REG IN	X OUT
00	-	-	-	-	-	D02	01	05
01	D06	04	D08	01	10, 11	D03	01	05
02	D06	03	E06	01	10, 11	D04	01	05
03	D06	02	E07	01	10, 11	D05	01	05
04						D09	01	05
05						D10	01	05
06						D11	01	05
07						E02	01	05
08						E03	01	05
09						E04	01	05
10						E05	01	05
11						E09	01	05
12						E10	01	05
13						E11	01	05
14						E12	01	05
15						D02	10	13
16						D03	10	13
17						D04	10	13
18						D05	10	13
19						D09	10	13
20						D10	10	13
21						D11	10	13
22						E02	10	13
23						E03	10	13
24						E04	10	13
25						E05	10	13
26						E09	10	13
27						E10	10	13
28						E11	10	13
29						E12	08	07
30						D02	08	07
31						D03	08	07
32						D04	08	07
33						D05	08	07
34						D09	08	07
35						D10	08	07
36						D11	08	07
37						E02	08	07
38						E03	08	07
39						E04	08	07
40						E05	08	07
41						E09	08	07
42						E10	08	07
43						E11	08	07
44						E12	08	07
45						D02	03	06
46						D03	03	06
47						D04	03	06
48						D05	03	06
49						D09	03	06
50						D10	03	06
51						D11	03	06
52						E02	03	06
53						E03	03	06
54						E04	03	06
55						E05	03	06
56						E09	03	06
57						E10	03	06
58						E11	03	06
59						E12	03	06

TABLE 5-2-3. CPU 3.2 KEY TEST POINTS (cont.)

	GK		CA			FR		
BIT NO	PAK LOC	$\overline{AA}$	PAK LOC	AA IN	$\overline{AA}$ OUT	PAK LOC	A(REG) IN	A OUT
00						B02	03	06
01	D06	04	C05	08	10/11	B02	01	05
02	D06	03	C05	03	01/02	B02	10	13
03	D06	02	C05	06	04	B02	08	07
04						B03	03	06
05						B03	01	05
06						B03	10	13
07						B03	08	07
08						B04	03	06
09						B04	01	05
10						B04	10	13
11						B04	08	07
12						B08	03	06
13						B08	01	05
14						B08	10	13
15						B08	08	07
16						B09	03	06
17						B09	01	05

	GK		CA			FR		
BIT NO	PAK LOC	$\overline{AB}$	PAK LOC	AB	AB	PAK LOC	B(REG) IN	B OUT
00						C02	03	06
01	C07	04	B07	08	10/11	C02	01	05
02	C07	03	B07	03	01/02	C02	10	13
03	C07	02	B07	06	04	C02	08	07
04						C03	03	06
05						C03	01	05
06						C03	10	13
07						C03	08	07
08						C04	03	06
09						C04	01	05
10						C04	10	13
11						C04	08	07
12						C08	03	06
13						C08	01	05
14						C08	10	13
15						C08	08	07
16						C09	03	06
17						C09	01	05



CONTROL DATA CANADIAN DEVELOPMENT DIVISION

X, A, B REGISTERS
CENTRAL PROCESSOR UNIT

CODE IDENT.		DWG. NO.	REV
34570	D	19981800	K
PUB. NO.		SHEET CPU 3-2	PAGE NO. 5-2-7

CARLETON INSTRUMENTS 138-3 200-13-74

DETAILED PAK DIAGRAM (CPU 3.3 & 3.4)

P, RA, MA, FL, RAE, FLE, EM/EE;
K1, K2 REGISTERS;
I0, I1, I49 SELECTORS

P REGISTER (PROGRAM ADDRESS REGISTER)

The P register is an 18-bit register that contains the program address of a 60-bit instruction word currently being executed. The initial contents of P are provided by the exchange jump package and incremented by +1 for each program step. The contents of P are modified by the addition of RA (P + RA) to determine the central memory location for each instruction word.

RA REGISTER (REFERENCE ADDRESS REGISTER)

The RA register contains a predetermined reference CM starting address for the current program in progress. RA is added to the address before each CM read or write reference, thus allowing multiprogramming and relocation of programs in central memory.

MA REGISTER (MONITOR ADDRESS REGISTER)

The MA register contains the absolute starting address of an exchange package that is used when executing a central exchange jump instruction, or when honoring a MAN exchange request from PPS if the monitor flag (MF) is clear.

FL REGISTER (FIELD LENGTH REGISTER)

The FL register is used to define the program field size. Before RA is added to an address for central memory, the address in the F register is checked against FL to determine if the upper limit of the program has been exceeded.

RAE REGISTER (REFERENCE ADDRESS ECS REGISTER)

The RAE register contains a 21-bit relative starting address for ECS. The lower 6 bits of RAE are always zero.

FLE REGISTER (FIELD LENGTH ECS REGISTER)

The 24-bit FLE register is used to define the program field size for ECS. The lower 6 bits of FLE are always zero.

The RAE and FLE registers serve the same purpose for ECS as do RA and FL for CM.

EM/EE REGISTERS (EXIT MODE, ERROR EXIT REGISTERS)

The EM register contains the exit mode selections for a program in operation. The exit mode bits control CPU action if the corresponding error is detected. Six exit mode bits are provided as follows:

Mode Selection Bit	Condition Sensed
48	Address out of range
49	Operand out of range
50	Indefinite operand
57	Parity error in ECS flag register operation
58	CMC input error
59	CM data error

The EE register is set by the actual error conditions sensed by the CPU or sent to the CPU from CMC. The error condition signals which set the respective EE bits are as follows:

Error Exit Bit	Condition
AORI - 48	Address out of range
INFOPR - 49	Operand out of range (infinite operand)
INDF - 50	Indefinite operand
FLGPAR - 57	Parity error in ECS flag register operation
PARERR - 58	CMC input error
DB+RDP - 59	CM data error

K1 REGISTER (SOURCE FIELD CM ADDRESS REGISTER)

The K1 register is used exclusively by compare/move instructions to specify the source field CM address.

K2 REGISTER (RESULT FIELD CM ADDRESS REGISTER)

The K2 register is used exclusively by compare/move instructions to specify the result or source field CM address.

P, RA, MA, FL LOGIC LAYOUT

The P, RA, MA and FL registers are contained on the FP modules. The CR9 register output provides an input path to P, RA, MA and FL during an exchange jump. A second input is provided to the P register from the F register. This path is used to store the updated contents of P back into the P register during a full RNI or branch, and to feed the address range test circuit also located on the FP module.

I0 SELECTOR

The I0 selector circuit provides input selection of the P, RA, MA, FL, K1 and K2 registers. The I0 selection code determines which of the six registers will be selected through I0. The I0 output feeds the data transmitters for storing P, RA, MA and FL during an exchange jump and provides an input path via I3 to the small adder.

RAE, FLE LOGIC LAYOUT

The RAE and FLE registers are contained on the FQ module. The CR9 register output provides an input path to RAE and FLE during an exchange jump.

I1 SELECTOR

The I1 selector circuit provides output selection of the RAE and FLE registers. The I1 output feeds the data transmitters for storing RAE and FLE during an exchange jump, and provides an input path via I15 and I5 to the large adder.

K1, K2 LOGIC LAYOUT

The K1 and K2 registers are contained on the JK module. The CR9 register output provides an input path to K1 and K2 via I49. CR9 is normally selected through I49 to gate the K1 and K2 portions of an instruction word into their respective registers. The second input path to K1 and K2 provides gating the updated K1 or K2 values from the F register back into the K1 or K2 registers during the address sequencing for a compare/move instruction.

EM, EE LOGIC LAYOUT

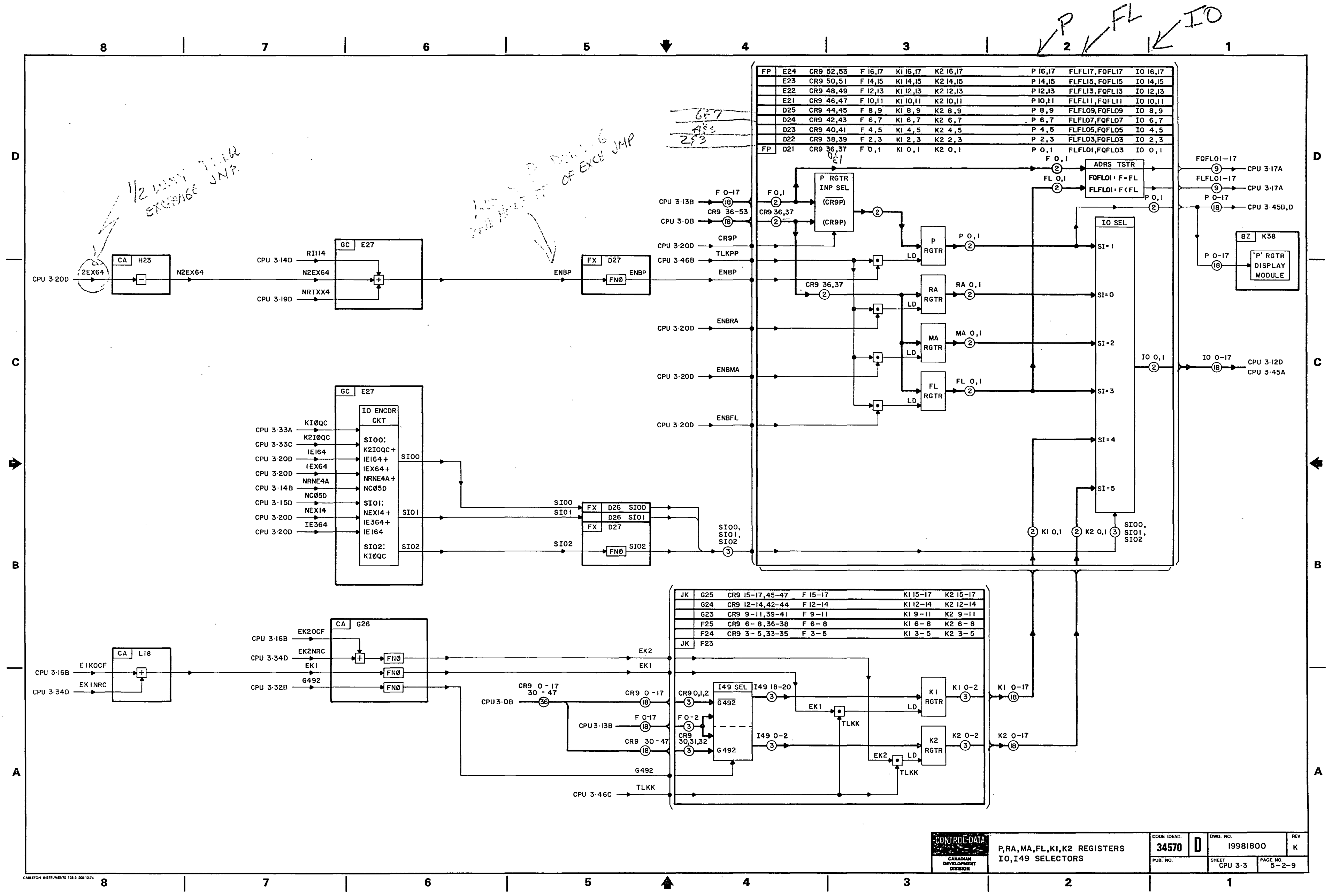
The EM and EE registers are contained on the FW module. The CR9 register output provides an input path to the EM register during an exchange jump. Inputs to the EE register consist of 3 error signals (AORI, INFOPR, INDF) generated within the CPU, and 3 error signals (DB+RDP, PARERR, FLGPAR) sent from the CMC.

The EM/EE registers feed the selector and error condition sense circuits. The selector circuit allows selection of the EM register to the data transmitters during an exchange jump, or of the EE register during an error exit sequence. The condition sense circuit compares each bit stored in the EE register with the exit mode bits in the EM register. When an error condition sets the respective EE register bit, and the exit mode bit in the EM register is also set, the ECONDS signal is generated to enable the return jump error exit sequence.

TABLE 5-2-4. CPU 3.3 KEY TEST POINTS

	FP (P, RA, MA, FL REGISTERS)						
BIT NO	PAK LOC	F (IN)	P (IN)	RA (OUT)	MA (OUT)	FL (OUT)	IO (OUT)
00	D21	08	14	05	06	10	01
01	D21	09	13	04	07	12	02
02	D22	08	14	05	06	10	01
03	D22	09	13	04	07	12	02
04	D23	08	14	05	06	10	01
05	D23	09	13	04	07	12	02
06	D24	08	14	05	06	10	01
07	D24	09	13	04	07	12	02
08	D25	08	14	05	06	10	01
09	D25	09	13	04	07	12	02
10	E21	08	14	05	06	10	01
11	E21	09	13	04	07	12	02
12	E22	08	14	05	06	10	01
13	E22	09	13	04	07	12	02
14	E23	08	14	05	06	10	01
15	E23	09	13	04	07	12	02
16	E24	08	14	05	06	10	01
17	E24	09	13	04	07	12	02

	JK (K REGISTER)	
BIT NO	PAK LOC	F (IN)
00	F23	11
01	F23	06
02	F23	07
03	F24	11
04	F24	06
05	F24	07
06	F25	11
07	F25	06
08	F25	07
09	G23	11
10	G23	06
11	G23	07
12	G24	11
13	G24	06
14	G24	07
15	G25	11
16	G25	06
17	G25	07



FP	E24	CR9 52,53	F 16,17	K1 16,17	K2 16,17	P 16,17	FLFL17,FQFL17	IO 16,17
	E23	CR9 50,51	F 14,15	K1 14,15	K2 14,15	P 14,15	FLFL15,FQFL15	IO 14,15
	E22	CR9 48,49	F 12,13	K1 12,13	K2 12,13	P 12,13	FLFL13,FQFL13	IO 12,13
	E21	CR9 46,47	F 10,11	K1 10,11	K2 10,11	P 10,11	FLFL11,FQFL11	IO 10,11
	D25	CR9 44,45	F 8,9	K1 8,9	K2 8,9	P 8,9	FLFL09,FQFL09	IO 8,9
	D24	CR9 42,43	F 6,7	K1 6,7	K2 6,7	P 6,7	FLFL07,FQFL07	IO 6,7
	D23	CR9 40,41	F 4,5	K1 4,5	K2 4,5	P 4,5	FLFL05,FQFL05	IO 4,5
	D22	CR9 38,39	F 2,3	K1 2,3	K2 2,3	P 2,3	FLFL03,FQFL03	IO 2,3
FP	D21	CR9 36,37	F 0,1	K1 0,1	K2 0,1	P 0,1	FLFL01,FQFL03	IO 0,1

JK	G25	CR9 15-17,45-47	F 15-17	K1 15-17	K2 15-17
	G24	CR9 12-14,42-44	F 12-14	K1 12-14	K2 12-14
	G23	CR9 9-11,39-41	F 9-11	K1 9-11	K2 9-11
	F25	CR9 6-8,36-38	F 6-8	K1 6-8	K2 6-8
	F24	CR9 3-5,33-35	F 3-5	K1 3-5	K2 3-5
JK	F23				

CONTROL DATA CANADIAN DEVELOPMENT DIVISION

P,RA,MA,FL,K1,K2 REGISTERS
IO,I49 SELECTORS

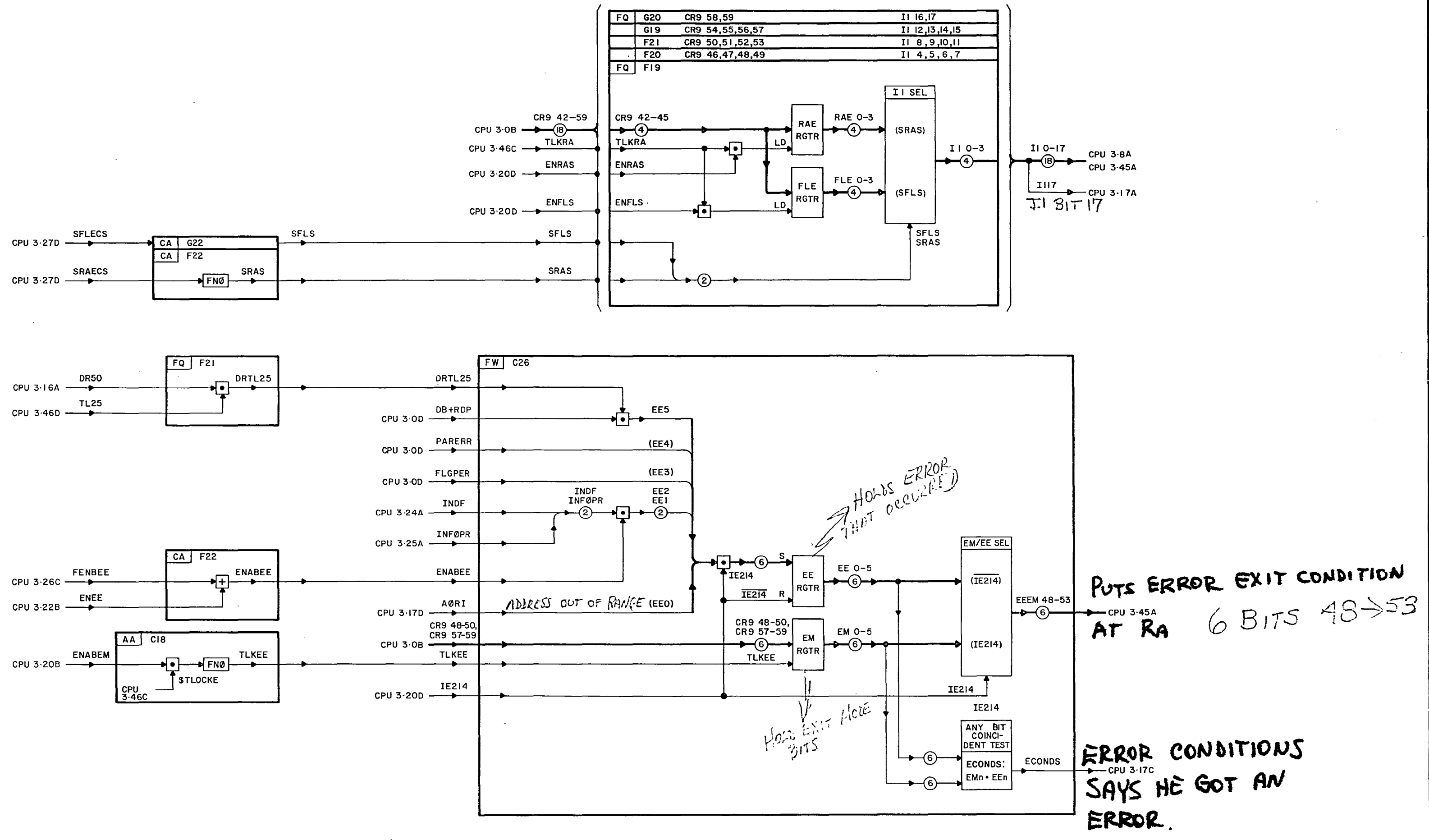
CODE IDENT.		DWG. NO.	REV
34570	D	19981800	K
PUB. NO.		SHEET CPU 3·3	PAGE NO. 5-2-9

CARLETON INSTRUMENTS 138-3 200-12-74

TABLE 5-2-5. CPU 3.4 KEY TEST POINTS

	FQ (RAE, FLE REGISTERS)				
BIT NO IN	PAK LOC	CR9 (IN)	RAE OUT	FLE OUT	BIT NO OUT
42	F19	-	04	02	00
43	F19	-	05	01	01
44	F19	14	08	11	02
45	F19	13	09	10	03
46	F20	-	04	02	04
47	F20	-	05	01	05
48	F20	14	08	11	06
49	F20	13	09	10	07
50	F21	-	04	02	08
51	F21	-	05	01	09
52	F21	14	08	11	10
53	F21	13	09	10	11
54	G19	-	04	02	12
55	G19	-	05	01	13
56	G19	14	08	11	14
57	G20	14	08	11	15
58	G20	-	04	02	16
59	G20	-	05	01	17

	FW		
BIT NO	PAK LOC	CR9 (IN)	EE (OUT)
48	C26	14	05
49	C26	12	06
50	C26	11	07
51	C26	-	10
52	C26	-	09
53	C26	-	08
54	-	-	-
55	-	-	-
56	-	-	-
57	C26	01	-
58	C26	03	-
59	C26	04	-



CONTROL DATA
CANADIAN DEVELOPMENT DIVISION

RAE,FLE,EE,EM REGISTERS
II,EM/EE SELECTORS

CODE IDENT.		DWG. NO.	REV
34570	D	19981800	K
PUB. NO.		SHEET CPU 3-4	PAGE NO. 5-2-11

CARLETON INSTRUMENTS 138-3 300-12-74

DETAILED PAK DIAGRAM (CPU 3.5)
D REGISTER, D ADDER, I14, I4

D REGISTER

The D register and C register together serve as input feeders to the D adder. The D register also functions as the output register for results from the D adder. I4 provides a 108-bit input to the D register from selector I14.

I14 SELECTOR

The I14 selector provides 108-bit selection of data from the D register or D adder to the I4 selector. I14S controls selection of the D register output through I14 to I4. The absence of I14S allows the D adder output to be gated through I14 to I4. The generation of I14S is controlled by FDI14 from the FAD/FMD sequence controls (CPU 3.26). FDI14 at HC module H14 generates I14S if $\overline{\text{CRY107}}$ or $\overline{44 + 45}$.

I4 SELECTOR

The I4 selector provides 108-bit selection of data from I14 to the D register. Signals I40 and I41 control selection through I4. Four transfers are possible:

1. I14 → I4 (No shift)
2. I14 → I4 (Right shift one)
3. I14 → I4 (Left shift one)
4. 0's → I4

The right and left shift capabilities are internal to I4. The right shift is end off, no sign extension; the left shift is not end around. The generation of I40 and I41 is controlled by the FAD/FMD sequence controls (CPU 3.26). I40 and I41 are also generated every common time 64 to gate 0's to the D register.

D ADDER

The D adder consists of a high speed arithmetic logic unit (ALU) capable of performing both arithmetic and logical functions. Arithmetic logic operations are selected by the DAS 0-3, DA-M signals. Group carry propagate (PG0003 - PG4107) and carry generate (GG0003 - GG4107) signals from the D-ALU are sent to the large adder first stage carry look-ahead control (CPU 3.6). The first stage carry look-ahead (FA modules H06, H11, H15, I06, I12, I13, I19) provides internal carry signals (CN0003 - CN4107) back to the D-ALU.

In its normal operation, the D adder performs a ones complement full add operation for: Boolean instructions (CPU 3.23), ECS instructions (CPU 3.27), integer sum/difference instructions and floating point instructions controlled by the FAD/FMD sequences (CPU 3.24, 3.25, 3.26).

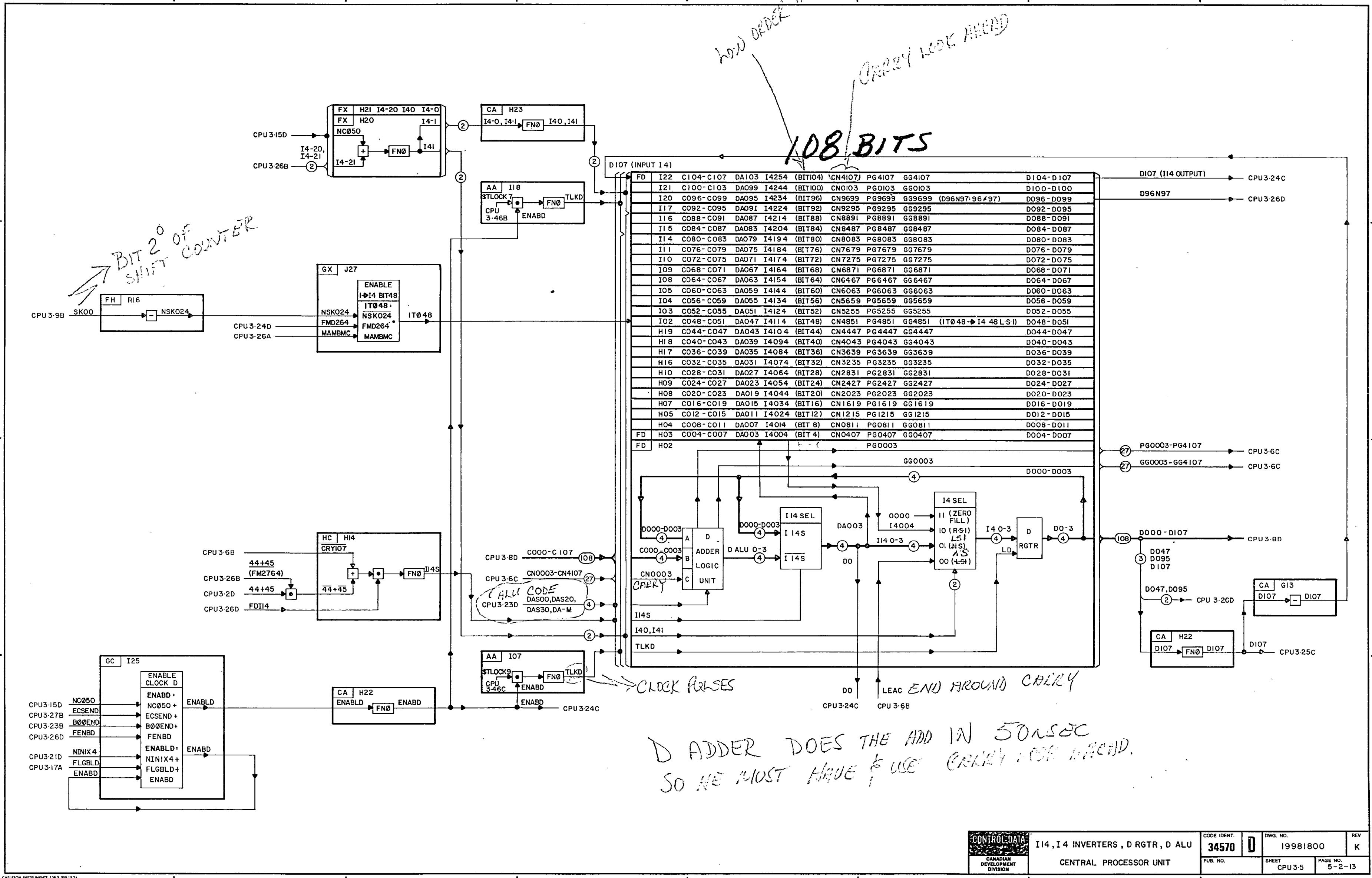
The D adder also performs logical operations (inclusive OR, exclusive OR, logical AND) using 60-bit operands for Boolean instructions (CPU 3.23).

TABLE 5-2-6. CPU 3.5 KEY TEST POINTS

	FD (D REGISTER + ALU)			
BIT NO.	PAK LOC	C (IN)	CN (IN)	D (OUT)
00	H02	13	14	03
01	H02	09	-	08
02	H02	10	-	12
03	H02	11	-	01
04	H03	13	14	03
05	H03	09	-	08
06	H03	10	-	12
07	H03	11	-	01
08	H04	13	14	03
09	H04	09	-	08
10	H04	10	-	12
11	H04	11	-	01
12	H05	13	14	03
13	H05	09	-	08
14	H05	10	-	12
15	H05	11	-	01
16	H07	13	14	03
17	H07	09	-	08
18	H07	10	-	12
19	H07	11	-	01
20	H08	13	14	03
21	H08	09	-	08
22	H08	10	-	12
23	H08	11	-	01
24	H09	13	14	03
25	H09	09	-	08
26	H09	10	-	12
27	H09	11	-	01
28	H10	13	14	03
29	H10	09	-	08
30	H10	10	-	12
31	H10	11	-	01
32	H16	13	14	03
33	H16	09	-	08
34	H16	10	-	12
35	H16	11	-	01

	FD			
BIT NO.	PAK LOC	C (IN)	CN (IN)	D (OUT)
36	H17	13	14	03
37	H17	09	-	08
38	H17	10	-	12
39	H17	11	-	01
40	H18	13	14	03
41	H18	09	-	08
42	H18	10	-	12
43	H18	11	-	01
44	H19	13	14	03
45	H19	09	-	08
46	H19	10	-	12
47	H19	11	-	01
48	I02	13	14	03
49	I02	09	-	08
50	I02	10	-	12
51	I02	11	-	01
52	I03	13	14	03
53	I03	09	-	08
54	I03	10	-	12
55	I03	11	-	01
56	I04	13	4	03
57	I04	09	-	08
58	I04	10	-	12
59	I04	11	-	01
60	I05	13	14	03
61	I05	09	-	08
62	I05	10	-	12
63	I05	11	-	01
64	I08	13	14	03
65	I08	09	-	08
66	I08	11	-	12
67	I08	10	-	01
68	I09	13	14	03
69	I09	09	-	08
70	I09	11	-	12
71	I09	10	-	01

	FD			
BIT NO.	PAK LOC	C (IN)	CN (IN)	D (OUT)
72	I10	13	14	03
73	I10	09	-	08
74	I10	11	-	12
75	I10	10	-	01
76	I11	13	14	03
77	I11	09	-	08
78	I11	11	-	12
79	I11	10	-	01
80	I14	13	14	03
81	I14	09	-	08
82	I14	11	-	12
83	I14	10	-	01
84	I15	13	14	03
85	I15	09	-	08
86	I15	11	-	12
87	I15	10	-	01
88	I16	13	14	03
89	I16	09	-	08
90	I16	11	-	12
91	I16	10	-	01
92	I17	13	14	03
93	I17	09	-	08
94	I17	11	-	12
95	I17	10	-	01
96	I20	13	14	03
97	I20	09	-	08
98	I20	11	-	12
99	I20	10	-	01
100	I21	13	14	03
101	I21	09	-	08
102	I21	11	-	12
103	I21	10	-	01
104	I22	13	14	03
105	I22	09	-	08
106	I22	11	-	12
107	I22	10	-	01



FD	I22	C104-C107	DA103	I4254	(BIT104)	CN4107	PG4107	GG4107		D104-D107
	I21	C100-C103	DA099	I4244	(BIT100)	CN0103	PG0103	GG0103		D100-D100
	I20	C096-C099	DA095	I4234	(BIT96)	CN9699	PG9699	GG9699	(D96N97·96≠97)	D096-D099
	I17	C092-C095	DA091	I4224	(BIT92)	CN9295	PG9295	GG9295		D092-D095
	I16	C088-C091	DA087	I4214	(BIT88)	CN8891	PG8891	GG8891		D088-D091
	I15	C084-C087	DA083	I4204	(BIT84)	CN8487	PG8487	GG8487		D084-D087
	I14	C080-C083	DA079	I4194	(BIT80)	CN8083	PG8083	GG8083		D080-D083
	I11	C076-C079	DA075	I4184	(BIT76)	CN7679	PG7679	GG7679		D076-D079
	I10	C072-C075	DA071	I4174	(BIT72)	CN7275	PG7275	GG7275		D072-D075
	I09	C068-C071	DA067	I4164	(BIT68)	CN6871	PG6871	GG6871		D068-D071
	I08	C064-C067	DA063	I4154	(BIT64)	CN6467	PG6467	GG6467		D064-D067
	I05	C060-C063	DA059	I4144	(BIT60)	CN6063	PG6063	GG6063		D060-D063
	I04	C056-C059	DA055	I4134	(BIT56)	CN5659	PG5659	GG5659		D056-D059
	I03	C052-C055	DA051	I4124	(BIT52)	CN5255	PG5255	GG5255		D052-D055
	I02	C048-C051	DA047	I4114	(BIT48)	CN4851	PG4851	GG4851	(ITØ48→I4 48 L·S·I)	D048-D051
	H19	C044-C047	DA043	I4104	(BIT44)	CN4447	PG4447	GG4447		D044-D047
	H18	C040-C043	DA039	I4094	(BIT40)	CN4043	PG4043	GG4043		D040-D043
	H17	C036-C039	DA035	I4084	(BIT36)	CN3639	PG3639	GG3639		D036-D039
	H16	C032-C035	DA031	I4074	(BIT32)	CN3235	PG3235	GG3235		D032-D035
	H10	C028-C031	DA027	I4064	(BIT28)	CN2831	PG2831	GG2831		D028-D031
	H09	C024-C027	DA023	I4054	(BIT24)	CN2427	PG2427	GG2427		D024-D027
	H08	C020-C023	DA019	I4044	(BIT20)	CN2023	PG2023	GG2023		D020-D023
	H07	C016-C019	DA015	I4034	(BIT16)	CN1619	PG1619	GG1619		D016-D019
	H05	C012-C015	DA011	I4024	(BIT12)	CN1215	PG1215	GG1215		D012-D015
	H04	C008-C011	DA007	I4014	(BIT 8)	CN0811	PG0811	GG0811		D008-D011
FD	H03	C004-C007	DA003	I4004	(BIT 4)	CN0407	PG0407	GG0407		D004-D007
FD	H02						PG0003			

CONTROL DATA CANADIAN DEVELOPMENT DIVISION

I14, I4 INVERTERS, D RGTR, D ALU

CENTRAL PROCESSOR UNIT

CODE IDENT.		DWG. NO.	REV
34570	D	19981800	K
PUB. NO.		SHEET CPU3·5	PAGE NO. 5-2-13

CARLETON INSTRUMENTS 138-3 200-13-74

DETAILED PAK DIAGRAM (CPU 3.6)
LARGE ADDER CARRY FUNCTION

The FA, FB and HC modules shown on CPU 3.6 depict the large adder carry look-ahead and carry functions. Group carry propagate (PG0003 - PG4107) and carry generate (GG0003 - GG4107) signals are sent from the D-ALU bits 0-107 to the first stage carry look-ahead control. This control consists of seven FA modules divided into two groups. The first group provides internal carry signals (CN0003 - CN0047) back to the D-ALU bits 0-47, and provides second stage group carry propagate (P10015, P11631, P13247) and carry generate (G10015, G11631, G13247) to the second stage carry look-ahead control for bits 0-47. The second group of first stage carry look-ahead FA modules provides internal carry signals (CN4851 - CN4107) back to the D-ALU bits 48-107, and provides second stage group carry propagate (P14863, P16479, P18095, P19607) and carry generate (G10015, G16479, G18095, G19607) to the second stage carry look-ahead control for bits 48-107.

The second stage carry look-ahead controls located on FA module H12 and HC module H14 provide group carry propagate and carry generate signals from the second stage carry look-ahead back to the first stage carry look-ahead controls. It also provides the end around carry signals (CRY047, CRY107) to the carry function control. This control, located on the FB module, provides the end around carry signals (LEAC, HEAC) to the first stage look-ahead, and distributes end around carry from bits 48-107 (LEAC) to the processor controls.

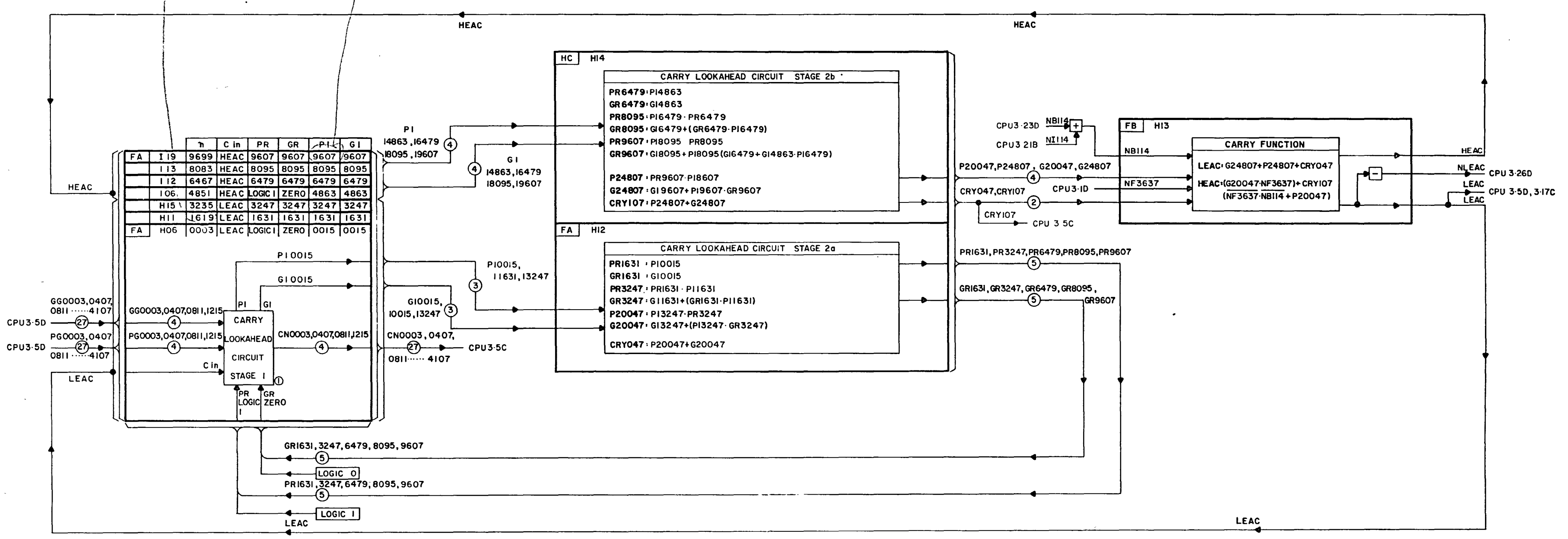
TABLE 5-2-7. CPU 3.6 KEY TEST POINTS

	FA		
	CARRY LOOKAHEAD STA. 1		
CARRY BIT NO.	PAK LOC	GG (IN)	PG (IN)
0003	H06	12	03
0407	H06	13	02
0811	H06	07	06
1215	H06	08	09
1619	H11	12	03
2023	H11	13	02
2427	H11	07	06
2831	H11	08	09
3235	H15	12	03
3639	H15	13	02
4043	H15	07	06
4447	H15	08	09
4851	I06	12	03
5255	I06	13	02
5659	I06	07	06
6063	I06	08	09
6467	I12	12	03
6871	I12	13	02
7275	I12	07	06
7679	I12	08	09
8083	I13	12	03
8487	I13	13	02
8891	I13	07	06
9295	I13	08	09
9699	I19	12	03
0103	I19	13	02
4107	I19	07	06

19

Bits $2^{16} \rightarrow 2$

BIT NUMBERS
Bits 96 thru 107



HEAC
HEAC
HEAC
LEAC
CPU3·5D
GG0003,0407, 0811 ······4107
CPU3·5D
PG0003,0407 0811 ······4107
GG0003,0407,0811,1215
PG0003,0407,0811,1215
P I 0015
G I 0015
CARRY LOOKAHEAD CIRCUIT STAGE I
Cin
PR LOGIC I
GR ZERO
CN0003,0407,0811,1215
P I 14863 ,16479 18095 ,19607
G I 14863,16479 18095,19607
P10015, 11631,13247
G10015, 10015,13247
CN0003 , 0407, 0811······ 4107
CPU3·5C
HC H14
CARRY LOOKAHEAD CIRCUIT STAGE 2b
PR6479 : PI4863
GR6479 : GI4863
PR8095 : PI6479 · PR6479
GR8095 : GI6479+(GR6479·PI6479)
PR9607 : PI8095 PR8095
GR9607 : GI8095 + PI8095(GI6479+GI4863·PI6479)
P24807 : PR9607·PI8607
G24807 : GI 9607+ PI9607·GR9607
CRY107 : P24807+G24807
FA H12
CARRY LOOKAHEAD CIRCUIT STAGE 2a
PRI631 : PI0015
GRI631 : GI0015
PR3247 : PRI631 · PI1631
GR3247 : GI1631+(GRI631·PI1631)
P20047 : PI3247·PR3247
G20047 : GI3247+(PI3247·GR3247)
CRY047 : P20047+G20047
CPU3·23D NB114
CPU3 21B NI114
P20047,P24807 , G20047,G24807
CRY047,CRY107
CPU3·1D
CRY107
CPU 3 5C
PRI631,PR3247,PR6479,PR8095,PR9607
GRI631,GR3247,GR6479,GR8095, GR9607
FB H13
NB114
NF3637
CARRY FUNCTION
LEAC : G24807+P24807+CRY047
HEAC : (G20047·NF3637)+ CRY107 (NF3637·NB114 + P20047)
HEAC
NLEAC CPU 3·26D
LEAC CPU 3·5D, 3·17C
LEAC
GRI631,3247,6479,8095,9607
LOGIC 0
PRI631,3247,6479,8095,9607
LOGIC I
LEAC
LEAC

		n	C in	PR	GR	P I	G I
FA	I 19	9699	HEAC	9607	9607	9607	9607
	I 13	8083	HEAC	8095	8095	8095	8095
	I 12	6467	HEAC	6479	6479	6479	6479
	I 06	4851	HEAC	LOGIC I	ZERO	4863	4863
	H15	3235	LEAC	3247	3247	3247	3247
	H11	1619	LEAC	1631	1631	1631	1631
FA	H06	0003	LEAC	LOGIC I	ZERO	0015	0015

NOTE ① LOGIC EQUATIONS FOR CARRY LOOKAHEAD CIRUIT STAGE I

Cn : (Cin · PR) + GR
Cn+k : (PGn · Cn) + GGn
Cn+2k : (PGn+k · Cn+k) + GGn+k
Cn+3k : (PGn+2k · Cn+k) + GGn+2k

PI : PGn · PGn+k · PGn+2k · PGn+3k
GI (GGn · PGn+k · PGn+2k · PGn+3k)
+ (GGn+k · PGn+2k · PGn+3k)
+ (GGn+2k · PGn+3k) + GGn+3k

k = 0404
values of n, Cin, PR, GR are specified in each pak location

CONTROL DATA CANADIAN DEVELOPMENT DIVISION	LARGE ADDER CARRY FUNCTION	CODE IDENT 34570	D	DWG NO 19981800 REV B
	CENTRAL PROCESSOR UNIT	PUB NO		SHEET CPU 3·6 PAGE NO 5-2-15

CARLETON INSTRUMENTS

10 MODULES
FOR CARRY LOOK AHEAD.

DETAILED PAK DIAGRAM (CPU 3.7)

I5 INVERTER CONTROL

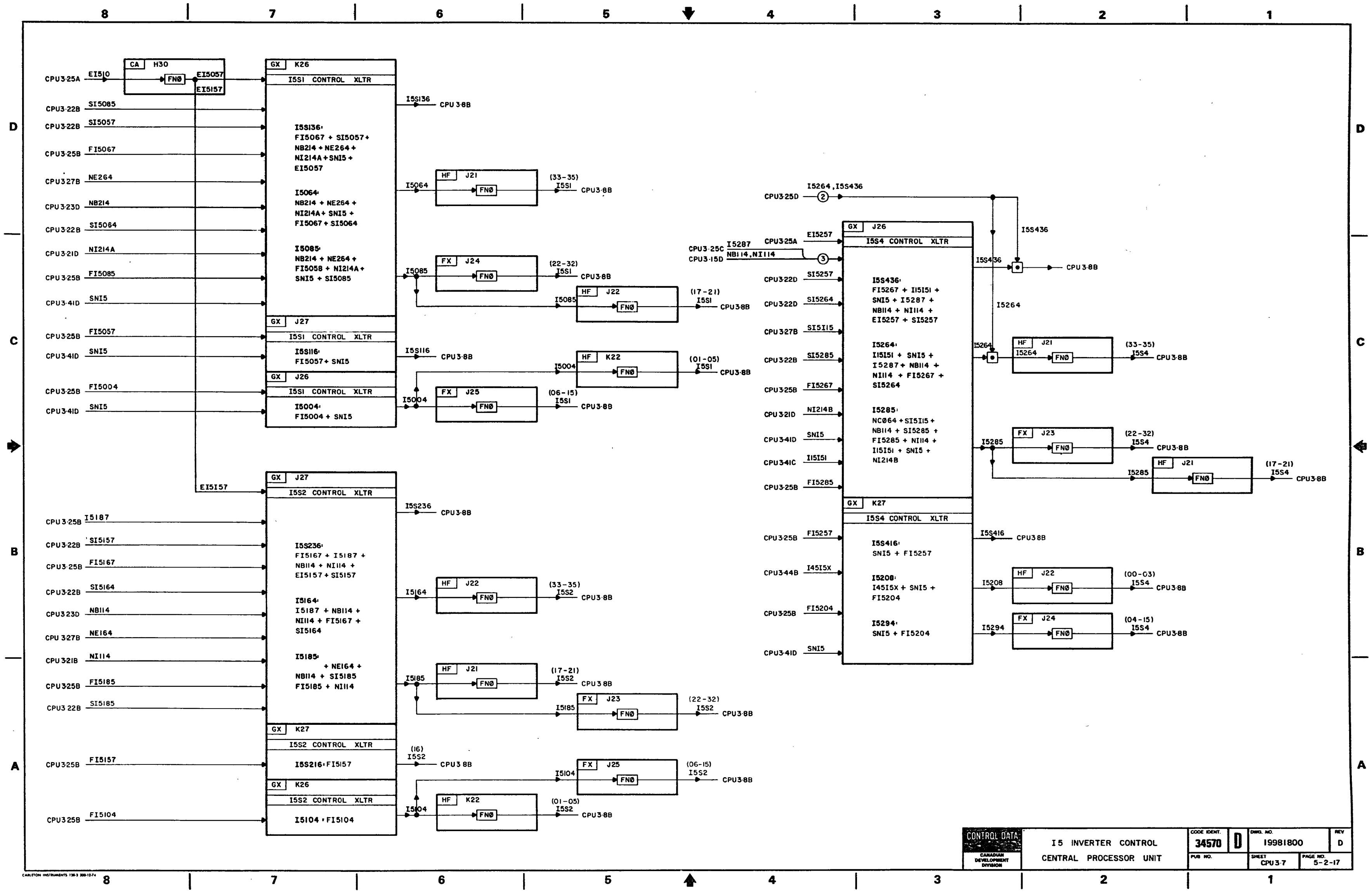
The I5 inverter rank controls the input to the C register. There are a number of varied inputs, not all of which are the same length as the 114-bit C register. For this reason, the I5 controls are broken into separate select networks for various areas of I5. Each select network provides a 3-bit code that will be translated to make the I5 input selection (CPU 3.8). Inputs to the select networks are from the instruction sequences. Outputs are defined in the following table:

I5 BITS	SELECT CODE		
	Bit 2	Bit 1	Bit 0
I5 105 - 107	I5S436	I5S236	I5S136
I5 96 - 104	I5264	I5164	I5064
I5 48 - 95	I5285	I5185	I5085
I5 45 - 47	I5S416	I5S216	I5S116
I5 9 - 44	I5294	I5104	I5004
I5 0 - 8	I5208	I5104	I5004

TABLE 5-2-8. CPU 3.7 KEY TEST POINTS

	GX		FX		HF	
SIGNAL	PAK LOC	T.P.	PAK LOC	T.P.	PAK LOC	T.P.
I5004	J26	1,2	J25	14	-	-
I5S116	J27	1,2	-	-	-	-
I5085	K26	12,13,14	J24	07	J22	05
I5064	K26	5,6,7	-	-	J21	13
I5S136	K26	3,4,11	-	-	-	-
I5104	K26	1,2	J25	07	-	-
I5S216	K27	1,2	-	-	-	-
I5185	J27	12,13,14	J23	14	J21	1,2*
I5164	J27	5,6,7	-	-	J22	1,2*
I5S236	J27	3,4,11	-	-	-	-
I5285	J26	12,13,14	J23	07	J21	5,7
I5264	J26	5,6,7	-	-	J21	09
I5S436	J26	3,4,11	-	-	-	-
I5294	K27	5,6,7	J24	14	-	-
I5208	K27	12,13,14	-	-	J22	09
I5S416	K27	3,4,11	-	-	-	-

*Signal complement at this test point.



8 7 6 5 4 3 2 1
D C B A
CPU3·25A EI510 CA H30 FNØ EI5057 EI5157
GX K26 I5S1 CONTROL XLTR
I5S136· FI5067 + SI5057 + NB214 + NE264 + NI214A + SNI5 + EI5057
I5064· NB214 + NE264 + NI214A + SNI5 + FI5067 + SI5064
I5085· NB214 + NE264 + FI5058 + NI214A + SNI5 + SI5085
CPU3·22B SI5085
CPU3·22B SI5057
CPU3·25B FI5067
CPU3·27B NE264
CPU3·23D NB214
CPU3·22B SI5064
CPU3·21D NI214A
CPU3·25B FI5085
CPU3·41D SNI5
I5S136 CPU3·8B
HF J21 FNØ (33-35) I5S1 CPU3·8B
FX J24 FNØ (22-32) I5S1 CPU3·8B
HF J22 FNØ (17-21) I5S1 CPU3·8B
GX J27 I5S1 CONTROL XLTR
I5S116· FI5057 + SNI5
CPU3·25B FI5057
CPU3·41D SNI5
I5S116 CPU3·8B
HF K22 FNØ (01-05) I5S1 CPU3·8B
GX J26 I5S1 CONTROL XLTR
I5004· FI5004 + SNI5
CPU3·25B FI5004
CPU3·41D SNI5
FX J25 FNØ (06-15) I5S1 CPU3·8B
GX J27 I5S2 CONTROL XLTR EI5157
I5S236· FI5167 + I5187 + NB114 + NI114 + EI5157 + SI5157
I5164· I5187 + NB114 + NI114 + FI5167 + SI5164
I5185· + NE164 + NB114 + SI5185 FI5185 + NI114
CPU3·25B I5187
CPU3·22B SI5157
CPU3·25B FI5167
CPU3·22B SI5164
CPU3·23D NB114
CPU3·27B NE164
CPU3·21B NI114
CPU3·25B FI5185
CPU3·22B SI5185
I5S236 CPU3·8B
HF J22 FNØ (33-35) I5S2 CPU3·8B
HF J21 FNØ (17-21) I5S2 CPU3·8B
FX J23 FNØ (22-32) I5S2 CPU3·8B
GX K27 I5S2 CONTROL XLTR
I5S216· FI5157
CPU3·25B FI5157
(16) I5S2 CPU3·8B
GX K26 I5S2 CONTROL XLTR
I5104 · FI5104
CPU3·25B FI5104
FX J25 FNØ (06-15) I5S2 CPU3·8B
HF K22 FNØ (01-05) I5S2 CPU3·8B
CPU3·25D I5264, I5S436
CPU3·25C I5287
CPU3·15D NB114, NI114
GX J26 I5S4 CONTROL XLTR
I5S436· FI5267 + I15151 + SNI5 + I5287 + NB114 + NI114 + EI5257 + SI5257
I5264· I15151 + SNI5 + I5287 + NB114 + NI114 + FI5267 + SI5264
I5285· NC064 + SI5115 + NB114 + SI5285 + FI5285 + NI114 + I15151 + SNI5 + NI214B
CPU3·25A EI5257
CPU3·22D SI5257
CPU3·22D SI5264
CPU3·27B SI5115
CPU3·22B SI5285
CPU3·25B FI5267
CPU3·21D NI214B
CPU3·41D SNI5
CPU3·41C I15151
CPU3·25B FI5285
I5S436 CPU3·8B
I5264
HF J21 FNØ (33-35) I5S4 CPU3·8B
FX J23 FNØ (22-32) I5S4 CPU3·8B
HF J21 FNØ (17-21) I5S4 CPU3·8B
GX K27 I5S4 CONTROL XLTR
I5S416· SNI5 + FI5257
I5208· I45I5X + SNI5 + FI5204
I5294· SNI5 + FI5204
CPU3·25B FI5257
CPU3·44B I45I5X
CPU3·25B FI5204
CPU3·41D SNI5
I5S416 CPU3·8B
HF J22 FNØ (00-03) I5S4 CPU3·8B
FX J24 FNØ (04-15) I5S4 CPU3·8B

CONTROL DATA CANADIAN DEVELOPMENT DIVISION

I5 INVERTER CONTROL

CENTRAL PROCESSOR UNIT

CODE IDENT.		DWG. NO.	REV
34570	D	19981800	D
PUB NO.	SHEET	PAGE NO.	
	CPU3-7	5-2-17	

DETAILED PAK DIAGRAM (CPU 3.8)

C, H REGISTERS; I15, I5 SELECTORS; I5 COMPLEMENT CONTROL

C REGISTER

The C register is 114 bits in size. Bits 0-107 serve as one of the input feeders to the D adder. Bits 108-113 are used by compare/move operations to catch the last character shifted from bit positions 48-107. The C register also provides a general path for data distribution to the processor. Its outputs feed the following circuits:

1.	D adder	CPU 3.5
2.	R Register	CPU 3.28
3.	I30 Selector	CPU 3.28
4.	Shift Network Rank 1	CPU 3.10
5.	Normalize Network	CPU 3.10

I5 SELECTOR

The I5 selector provides input selection of data from various sources within the processor to the I5 complement control. Inputs to I5 are received from the following circuits:

1.	D Register	CPU 3.15
2.	C Register	CPU 3.8
3.	X Register	CPU 3.2
4.	I15 Selector	CPU 3.8
5.	I45 Selector	CPU 3.31
6.	Shift Network Rank 4	CPU 3.11

In addition to the above input selections, I5 can generate its own internal constants of 4_8, 6_8 or zeros to the I5 complement control. Input or constant selection is determined by a 3-bit code generated from the I5 control (CPU 3.7). Decoding of the selection code bits at I5 allows the following input or constant selections to be made:

Selection Code	Input or Constant Selection		
0	0's	→	$I5_{0-113}$
1	D Register	→	$I5_{0-107}$
2	C Register	→	$I5_{0-113}$
3	4_8's	→	$I5_{0-107}$
4	$I45_{0-6}$	→	$I5_{0-6}$
	$I15_{0-59}$	→	$I5_{48-107}$
5	SN4	→	$I5_{0-113}$
6	X Register	→	$I5_{48-107}$
7	6_8's	→	$I5_{0-107}$

I5 COMPLEMENT CONTROL

The I5 complement control allows the I5 output to be complemented before it is sent to the C register. Complement selection is controlled by a 2-bit complement code (I5C1, I5C2).

H REGISTER

The H register acts as a 60-bit holding register for compare/move operations. The H register outputs feed the I15 selector located on the same JA modules.

I15 SELECTOR

Selector I15 provides 60-bit input selection from the H register and 18-bit input selection from the F register and selector I1. Selector I15's outputs provide one of the 60-bit data inputs at selector I5 bit positions 48-107. I15's input selection is determined by selection control signals F46X, FEXI15, and ESI15. These control signals allow four input selections through I15:

Input Select Control	Input Selection		
1. $F46X \cdot \overline{FEXI15} \cdot \overline{ESI15}$	H register_{0-59}	→	$I15_{0-59}$
2. $\overline{F46X} \cdot FEXI15$	F register_{0-10}	→	$I15_{48-58}$
	C107	→	$I15_{59}$
3. $\overline{F46X} \cdot ESI15$	I1 selector_{0-17}	→	$I15_{6-23}$
4. $\overline{F46X} \cdot \overline{ESI15}$	F register_{0-17}	→	$I15_{0-17}$

F46X allows selection of H register bits 0-59 to I15 bits 0-59 for compare/move operations. FEXI15 allows gating of the biased exponent and coefficient sign (C107) from F register bits 0-10 to I15 bits 48-59. FEXI15 is enabled during shift and floating instructions to pack the computed exponent with the result coefficient. ESI15 allows gating of the I1 selector output to I15 bits 6-23 for ECS instructions. When ESI15 is selected, zeros are gated to I15 bit positions 0-5. $\overline{F46X} \cdot \overline{ESI15}$ allows gating of the F register bits 0-17 to I15 bits 0-17. This input selection is used by 7X increment, population count (47) and ECS instructions.

TABLE 5-2-9. CPU 3.8 KEY TEST POINTS

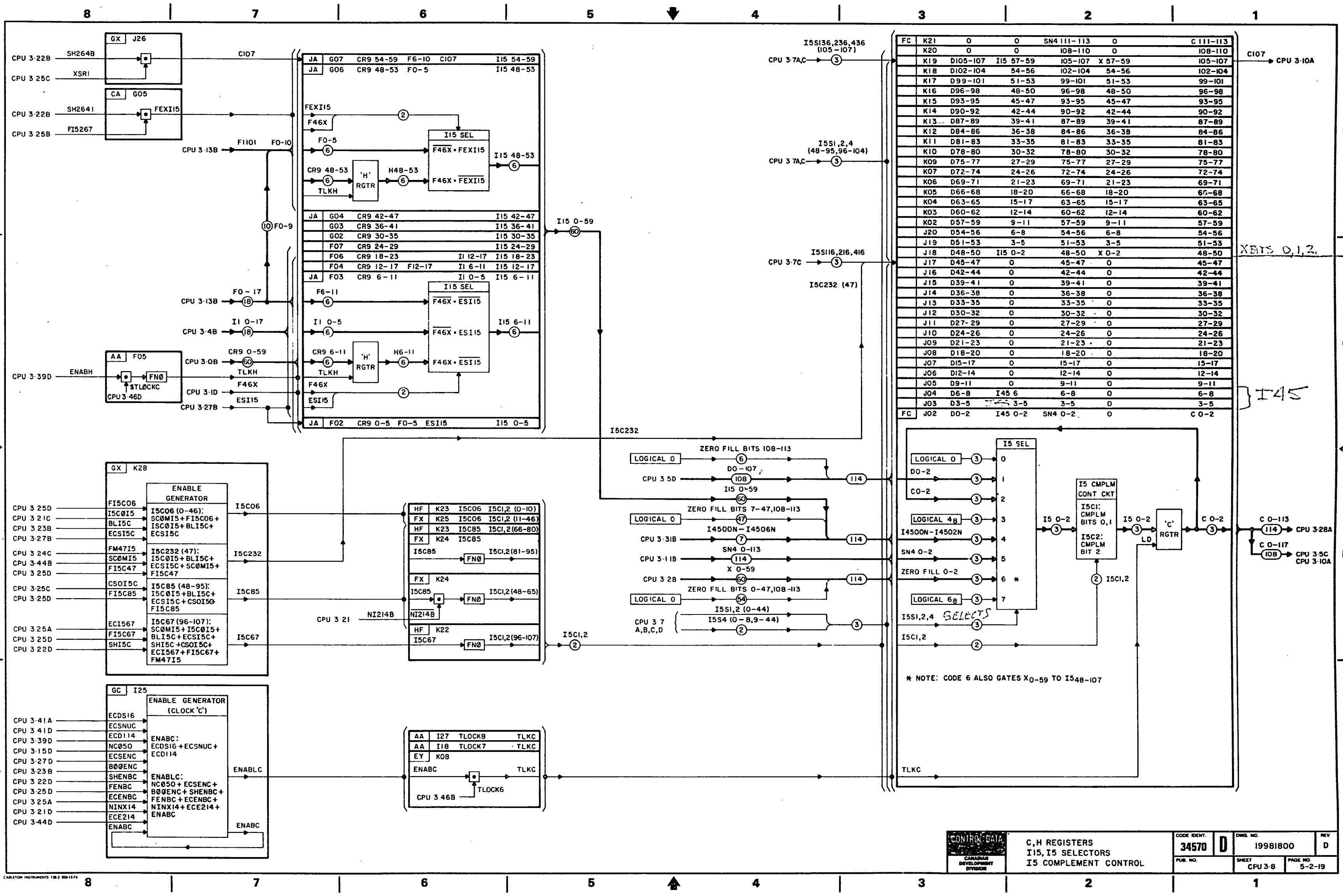
	JA (I15 SEL)			FC (I5 COMP CNTRL.)			FC	
BIT NO.	PAK LOC	CR9 (IN)	$\overline{I15}$ OUT	PAK LOC	SN4 (IN)	I5	PAK LOC	X (IN)
00	F02	5	3	J02	7	8	J18	2
01	F02	7	2	J02	5	10	J18	3
02	F02	8	1	J02	4	9	J18	1
03	F02	9	14	J03	7	8	J19	2
04	F02	10	12	J03	5	10	J19	3
05	F02	11	13	J03	4	9	J19	1
06	F03	5	3	J04	7	8	J20	2
07	F03	7	2	J04	5	10	J20	3
08	F03	8	1	J04	4	9	J20	1
09	F03	9	14	J05	7	8	K02	2
10	F03	10	12	J05	5	10	K02	3
11	F03	11	13	J05	4	9	K02	1
12	F04	5	3	J06	7	8	K03	2
13	F04	7	2	J06	5	10	K03	3
14	F04	8	1	J06	4	9	K03	1
15	F04	9	14	J07	7	8	K04	2
16	F04	10	12	J07	5	10	K04	3
17	F04	11	13	J07	4	9	K04	1
18	F06	5	3	J08	7	8	K05	2
19	F06	7	2	J08	5	10	K05	3
20	F06	8	1	J08	4	9	K05	1
21	F06	9	14	J09	7	8	K06	2
22	F06	10	12	J09	5	10	K06	3
23	F06	11	13	J09	4	9	K06	1
24	F07	5	3	J10	7	8	K07	2
25	F07	7	2	J10	5	10	K07	3
26	F07	8	1	J10	4	9	K07	1
27	F07	9	14	J11	7	8	K09	2
28	F07	10	12	J11	5	10	K09	3

	FC		
BIT NO.	PAK LOC	SN4 (IN)	I5
60	K03	7	8
61	K03	5	10
62	K03	4	9
63	K04	7	8
64	K04	5	10
65	K04	4	9
66	K05	7	8
67	K05	5	10
68	K05	4	9
69	K06	7	8
70	K06	5	10
71	K06	4	9
72	K07	7	8
73	K07	5	10
74	K07	4	9
75	K09	7	8
76	K09	5	10
77	K09	4	9
78	K10	7	8
79	K10	5	10
80	K10	4	9
81	K11	7	8
82	K11	5	10
83	K11	4	9
84	K12	7	8
85	K12	5	10
86	K12	4	9
87	K13	7	8
88	K13	5	10

TABLE 5-2-9. CPU 3.8 KEY TEST POINTS (cont.)

	JA (I15 SEL)			FC (I5 COMP CNTRL.)			FC	
BIT NO.	PAK LOC	CR9 (IN)	$\overline{I15}$ OUT	PAK LOC	SN4 (IN)	I5	PAK LOC	X (IN)
29	F07	11	13	J11	4	9	K09	1
30	G02	5	3	J12	7	8	K10	2
31	G02	7	2	J12	5	10	K10	3
32	G02	8	1	J12	4	9	K10	1
33	G02	9	14	J13	7	8	K11	2
34	G02	10	12	J13	5	10	K11	3
35	G02	11	13	J13	4	9	K11	1
36	G03	5	3	J14	7	8	K12	2
37	G03	7	2	J14	5	10	K12	3
38	G03	8	1	J14	4	9	K12	1
39	G03	9	14	J15	7	8	K13	2
40	G03	10	12	J15	5	10	K13	3
41	G03	11	13	J15	4	9	K13	1
42	G04	5	3	J16	7	8	K14	2
43	G04	7	2	J16	5	10	K14	3
44	G04	8	1	J16	4	9	K14	1
45	G04	9	14	J17	7	8	K15	2
46	G04	10	12	J17	5	10	K15	3
47	G04	11	13	J17	4	9	K15	1
48	G06	5	3	J18	7	8	K16	2
49	G06	7	2	J18	5	10	K16	3
50	G06	8	1	J18	4	9	K16	1
51	G06	9	14	J19	7	8	K17	2
52	G06	10	12	J19	5	10	K17	3
53	G06	11	13	J19	4	9	K17	1
54	G07	5	3	J20	7	8	K18	2
55	G07	7	2	J20	5	10	K18	3
56	G07	8	1	J20	4	9	K18	1
57	G07	9	14	K02	7	8	K19	2
58	G07	10	12	K02	5	10	K19	3
59	G07	11	13	K02	4	9	K19	1

	FC		
BIT NO.	PAK LOC	SN4 (IN)	I5
89	K13	4	9
90	K14	7	8
91	K14	4	10
92	K14	4	9
93	K15	7	8
94	K15	5	10
95	K15	4	9
96	K16	7	8
97	K16	5	10
98	K16	4	9
99	K17	7	8
100	K17	5	10
101	K17	4	9
102	K18	7	8
103	K18	5	10
104	K18	4	9
105	K19	7	8
106	K19	5	10
107	K19	4	9
108	K20	7	8
109	K20	5	10
110	K20	4	9
111	K21	7	8
112	K21	5	10
113	K21	4	9


8
7
6
5
4
3
2
1
D
C
B
A
GX J26
SH264B
XSRI
CPU 3·22B
CPU 3 25C
CI07
CA G05
SH2641
FI5267
CPU 3·22B
CPU 3 25B
FEXI15
CPU 3·13B
FII01
F0-10
JA G07 CR9 54-59 F6-10 CI07 I15 54-59
JA G06 CR9 48-53 F0-5 I15 48-53
F46X
F0-5
I15 SEL
F46X·FEXI15
F46X·FEXI15
I15 48-53
CR9 48-53
'H' RGTR
H48-53
TLKH
JA G04 CR9 42-47 I15 42-47
G03 CR9 36-41 I15 36-41
G02 CR9 30-35 I15 30-35
F07 CR9 24-29 I15 24-29
F06 CR9 18-23 I1 12-17 I15 18-23
F04 CR9 12-17 F12-17 I1 6-11 I15 12-17
JA F03 CR9 6-11 I1 0-5 I15 6-11
F0-9
F0 - 17
CPU 3·13B
F6-11
I1 0-17
CPU 3·4B
I1 0-5
I15 6-11
AA F05
ENABH
CPU 3·39D
FNØ
STLØCKC
CPU 3·46D
CPU 3·0B
CR9 0-59
CR9 6-11
H6-11
TLKH
CPU 3·1D
F46X
CPU 3·27B
ESI15
JA F02 CR9 0-5 F0-5 ESI15 I15 0-5
I15 0-59
I5C232
GX K28
ENABLE GENERATOR
CPU 3·25D FI5C06
CPU 3·21C I5CØI5
CPU 3·23B BLI5C
CPU 3·27B ECSI5C
CPU 3·24C FM47I5
CPU 3·44B SCØMI5
CPU 3·25D FI5C47
CPU 3·25C CSOI5C
CPU 3·25D FI5C85
CPU 3·25A ECI567
CPU 3·25D FI5C67
CPU 3·22D SHI5C
I5C06 (0-46): SCØMI5+FI5C06+ ISCØI5+BLI5C+ ECSI5C
I5C232 (47): I5CØI5+BLI5C+ ECSI5C+SCØMI5+ FI5C47
I5C85 (48-95): I5CØI5+BLI5C+ ECSI5C+CSOI5C+ FI5C85
I5C67 (96-107): SCØMI5+I5CØI5+ BLI5C+ECSI5C+ SHI5C+CSOI5C+ ECI567+FI5C67+ FM47I5
I5C06
I5C232
I5C85
I5C67
HF K23 I5C06 I5CI,2 (0-10)
FX K25 I5C06 I5CI,2 (11-46)
HF K23 I5C85 I5CI,2 (66-80)
FX K24 I5C85
I5C85 I5CI,2 (81-95)
FX K24
I5C85 I5CI,2 (48-65)
CPU 3 21 NI214B
HF K22
I5C67 I5CI,2 (96-107)
I5CI,2
GC I25
ENABLE GENERATOR (CLOCK 'C')
CPU 3·41A ECDSI6
CPU 3·41D ECSNUC
CPU 3·39D ECDI14
CPU 3·15D NCØ50
CPU 3·27D ECSENC
CPU 3·23B BØØENC
CPU 3·22D SHENBC
CPU 3·25D FENBC
CPU 3·25A ECENBC
CPU 3·21D NINXI4
CPU 3·44D ECE214
ENABC
ENABC: ECDSI6+ECSNUC+ ECDI14
ENABLC: NCØ50+ECSENC+ BØØENC+SHENBC+ FENBC+ECENBC+ NINXI4+ECE214+ ENABC
ENABLC
ENABC
AA I27 TLOCK8 TLKC
AA I18 TLOCK7 TLKC
EY K08
ENABC TLKC
CPU 3 46B
TLOCK6
I5SI36,236,436 (105-107)
CPU 3·7A,C
I5SI,2,4 (48-95,96-104)
CPU 3 7A,C
I5SII6,216,416
CPU 3·7C
I5C232 (47)
FC K21 0 0 SN4 III-II3 0 C III-II3
K20 0 0 I08-II0 0 I08-II0
KI9 DI05-I07 I15 57-59 I05-I07 X 57-59 I05-I07
KI8 DI02-I04 54-56 I02-I04 54-56 I02-I04
KI7 D99-I0I 51-53 99-I0I 51-53 99-I0I
KI6 D96-98 48-50 96-98 48-50 96-98
KI5 D93-95 45-47 93-95 45-47 93-95
KI4 D90-92 42-44 90-92 42-44 90-92
KI3 D87-89 39-41 87-89 39-41 87-89
KI2 D84-86 36-38 84-86 36-38 84-86
KII D81-83 33-35 81-83 33-35 81-83
KI0 D78-80 30-32 78-80 30-32 78-80
K09 D75-77 27-29 75-77 27-29 75-77
K07 D72-74 24-26 72-74 24-26 72-74
K06 D69-71 21-23 69-71 21-23 69-71
K05 D66-68 18-20 66-68 18-20 66-68
K04 D63-65 15-17 63-65 15-17 63-65
K03 D60-62 12-14 60-62 12-14 60-62
K02 D57-59 9-11 57-59 9-11 57-59
J20 D54-56 6-8 54-56 6-8 54-56
JI9 D51-53 3-5 51-53 3-5 51-53
JI8 D48-50 I15 0-2 48-50 X 0-2 48-50
JI7 D45-47 0 45-47 0 45-47
JI6 D42-44 0 42-44 0 42-44
JI5 D39-41 0 39-41 0 39-41
JI4 D36-38 0 36-38 0 36-38
JI3 D33-35 0 33-35 0 33-35
JI2 D30-32 0 30-32 0 30-32
JII D27-29 0 27-29 0 27-29
JI0 D24-26 0 24-26 0 24-26
J09 D21-23 0 21-23 0 21-23
J08 DI8-20 0 I8-20 0 I8-20
J07 DI5-I7 0 I5-I7 0 I5-I7
J06 DI2-I4 0 I2-I4 0 I2-I4
J05 D9-II 0 9-II 0 9-II
J04 D6-8 I45 6 6-8 0 6-8
J03 D3-5 ~~I45~~ 3-5 3-5 0 3-5
FC J02 D0-2 I45 0-2 SN4 0-2 0 C 0-2
CI07
CPU 3·10A
XBITS 0,1,2.
I45
LOGICAL 0
ZERO FILL BITS 108-113
D0-107
CPU 3 5D
I15 0-59
ZERO FILL BITS 7-47,108-113
LOGICAL 0
I4500N-I4506N
CPU 3·31B
SN4 0-113
CPU 3·11B
X 0-59
CPU 3 2B
ZERO FILL BITS 0-47,108-113
LOGICAL 0
I5SI,2 (0-44)
CPU 3 7 A,B,C,D
I5S4 (0-8,9-44)
I15 SEL
LOGICAL 0
D0-2
C0-2
LOGICAL 4₈
I4500N-I4502N
SN4 0-2
ZERO FILL 0-2
LOGICAL 6₈
I5SI,2,4
SELECTS
I5CI,2
I15 0-2
I5 CMPLM CONT CKT
I5CI: CMPLM BITS 0,1
I5C2: CMPLM BIT 2
I5 0-2
LD
'C' RGTR
C 0-2
C 0-113
CPU 3·28A
C 0-117
CPU 3·5C
CPU 3·10A
I5CI,2
TLKC
* NOTE: CODE 6 ALSO GATES X_{0-59} TO $I5_{48-107}$
CARLETON INSTRUMENTS

CONTROL DATA
CANADIAN DEVELOPMENT DIVISION

C,H REGISTERS
I15, I5 SELECTORS
I5 COMPLEMENT CONTROL

CODE IDENT.		DWG. NO.	REV
34570	D	19981800	D
PUB. NO.		SHEET CPU 3·8	PAGE NO. 5-2-19

SK COUNTER, I19, I9 SELECTORS

I19 SELECTOR

The I19 selector provides input selection of U3 register bits 0-5 or SCRX bits 1-5 through I19 to the I9 selector.

I9 SELECTOR

The I9 selector provides input selection to the SK counter. The following circuits are fed to the I9 selector:

1. Normalize count
2. F register
3. I19 selector

NOTE: These inputs are received in complement form because I9 always complements.

In addition, I9 can generate its own internal constants of 60_8 or 74_8 for iteration counts required by the FMD sequence.

Input and constant selections through I9 are controlled by a 2-bit selection code SLI90 and SLI91. The selection codes generated provide the following input or constant selections through I9:

Selection Code	Input or Constant Selection		
0 . COMT64	60_8	→	I9
0 . $\overline{\text{COMT64}}$	74_8	→	I9
1	Normalize Count	→	I9
2	I19	→	I9
3 . $\overline{\text{FL128}}$. $\overline{\text{FCOM}}$	F register	→	I9
3 . $\overline{\text{FL128}}$. FCOM	$\overline{\text{F register}}$	→	I9
FL128	1's	→	I9

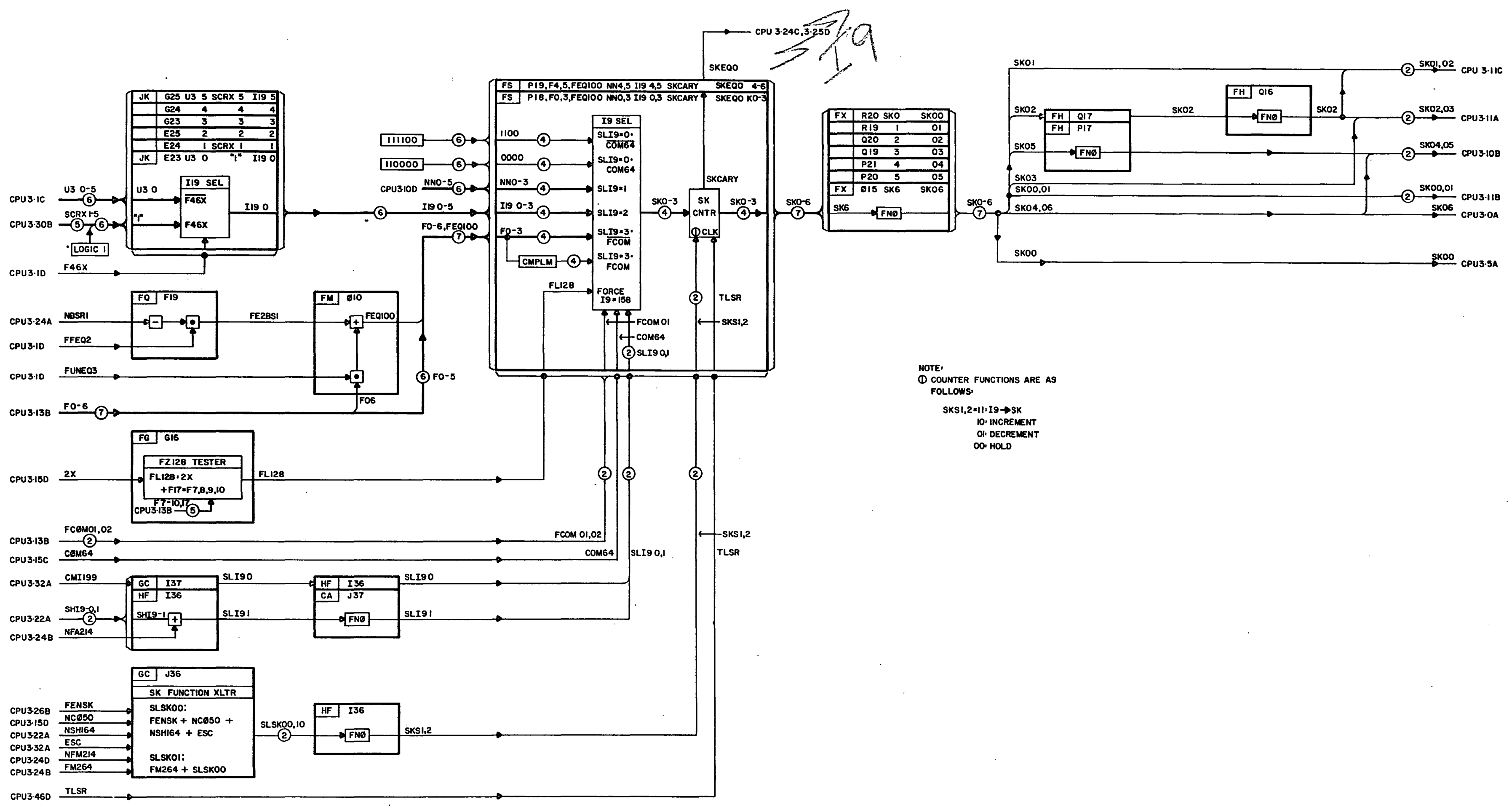
The F register is gated to I19 when a select code of 3 is generated. However, F may be complemented through I9 if the F register sign (F17) is negative.

SK REGISTER

The SK register serves a dual purpose. It acts as a register for storing the shift count, and as an iteration counter for the FMD sequence. The SKS1 and SKS2 signals control the functioning of the SK register. The shift or iteration count sent from I9 is stored in SK when a preset function is generated. During iterative steps of the FMD sequence, a decrement function code reduces the count by one for every clock pulse. When active, the SKEQ0 signal indicates that the iteration count has been decremented to zero.

TABLE 5-2-10. CPU 3.9 KEY TEST POINTS

	JK (I19 SEL)				FS (I9 SEL & SK)			FX			FH		FH	
BIT NO.	PAK LOC.	U3 (IN)	SCRX (IN)	I19	PAK LOC.	NN (IN)	SK (IN)	PAK LOC.	SK (IN)	SK (OUT)	PAK LOC.	SK (IN)	PAK LOC.	SK (IN)
00	F23	3	12	4	P18	9	2	R20	01	10, 11				
01	F24	3	12	4	P18	10	3	R19	01	10, 11				
02	F25	3	12	4	P18	11	4	Q20	01	10, 11	Q17	11	Q16	11
03	G23	3	12	4	P18	12	7	Q19	01	10, 11				
04	G24	3	12	4	P19	9	2	P21	01	10, 11				
05	G25	3	12	4	P19	10	3	P20	01	10, 11	P17	11		
06								O15	01	10, 11				



NOTE:
① COUNTER FUNCTIONS ARE AS FOLLOWS:

SKS1,2=11: I9→SK
10: INCREMENT
01: DECREMENT
00: HOLD

3-9 / I9

CONTROL DATA CANADIAN DEVELOPMENT DIVISION

SELECTOR I9, I19 SK COUNTER / CENTRAL PROCESSOR UNIT	CODE IDENT. 34570	D	DWG. NO. 19981800	REV A
	PUB. NO.		SHEET CPU3-9	PAGE NO. 5-2-21

CARLETON INSTRUMENTS

DETAILED PAK DIAGRAM (CPU 3.10)

SHIFT NETWORK RANKS 1 & 2
NORMALIZE NETWORK

SHIFT NETWORK RANK 1

The first rank of the shift network provides for right shifts of 64. The SK register bit 6 determines whether the C register output will be shifted by this first rank. The shift network first rank output feeds the second rank of the shift network.

SHIFT NETWORK RANK 2

The second rank of the shift network provides for right and left shifts of 16, 32 and 48. The SK register bits 4 and 5 determine whether the output from rank 1 (SN1 0-107) will be shifted by this second rank. RS determines the shift direction, left or right. The shift network second rank output feeds the third rank of the shift network.

HIGH/LOW SELECT CIRCUIT

The high/low select circuit allows selection of C register bits 0-47 or 48-95, plus bits 96-107, to the X register. This selection is required for use by double precision instructions and floating divide instructions.

NORMALIZE NETWORK

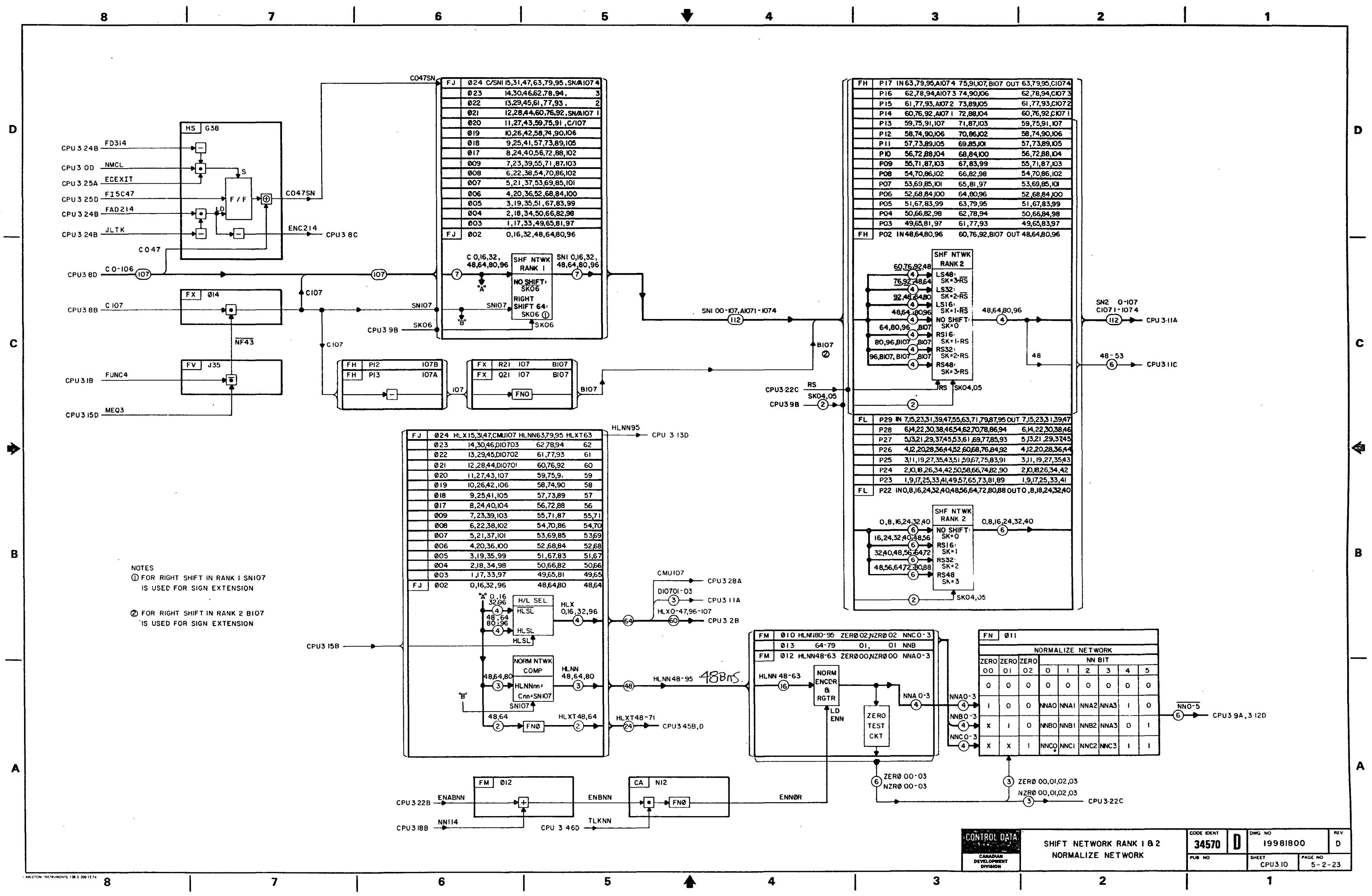
The normalize network receives its input from the 48-bit coefficient contained in the C register bits 48-95. The normalize network is a static network that forms at its output the 6-bit normalize count. This count is sent to the SK register via 19. The shift network, which is under control of SK, left shifts the coefficient the number of places specified by the normalize count, until the most significant 1 bit of the coefficient is in the bit 95 position.

Since the normalize network assumes a positive quantity, the circuits on the FJ modules compare each bit of the coefficient with the sign bit. This produces a 48-bit positive quantity that is sent to the normalize network where it is divided into three groups of 16 bits each. Each group generates a 4-bit count, giving the location of the highest order 1 bit in that group. The three groups of 4 bits (NNA 0-3, NNB 0-3, and NNC 0-3) are fed to the second stage of the normalize network where the 6-bit normalize count is formed.

The second stage of the normalize network also detects if a normalize count of zero is being formed, and automatically adds 60_8 to the count. This ensures a shift count of 48_{10} on a 24 instruction with a coefficient equal to zero.

TABLE 5-2-11. CPU 3.10 KEY TEST POINTS

	FJ		FH/FL			FJ		FH/FL			FJ		FH/FL	
BIT NO.	PAK LOC.	C (IN)	PAK LOC.	SN1 (IN)	BIT NO.	PAK LOC.	C (IN)	PAK LOC.	SN1 (IN)	BIT NO.	PAK LOC.	C (IN)	PAK LOC.	SN1 (IN)
00	O02	11	P22	07	38	O08	07	P28	06	76	O21	09	P14	08
01	O03	11	P23	07	39	O09	07	P29	06	77	O22	09	P15	08
02	O04	11	P24	07	40	O17	07	P22	08	78	O23	09	P16	08
03	O05	11	P25	07	41	O18	07	P23	08	79	O24	09	P17	08
04	O06	11	P26	07	42	O19	07	P24	08	80	O02	04	P02	09
05	O07	11	P27	07	43	O20	07	P25	08	81	O03	04	P03	09
06	O08	11	P28	07	44	O21	07	P26	08	82	O04	04	P04	09
07	O09	11	P29	07	45	O22	07	P27	08	83	O05	04	P05	09
08	O17	11	P22		46	O23	07	P28	08	84	O06	04	P06	09
09	O18	11	P23		47	O24	07	P29	08	85	O07	04	P07	09
10	O19	11	P24		48	O02	02	P02	04	86	O08	04	P08	09
11	O20	11	P25		49	O03	03	P03	04	87	O09	04	P09	09
12	O21	11	P26		50	O04	03	P04	04	88	O17	04	P10	09
13	O22	11	P27		51	O05	03	P05	04	89	O18	04	P11	09
14	O23	11	P28		52	O06	03	P06	04	90	O19	04	P12	09
15	O24	11	P29		53	O07	03	P07	04	91	O20	04	P13	09
16	O02	10	P22	05	54	O08	03	P08	04	92	O21	04	P14	09
17	O03	10	P23	05	55	O09	03	P09	04	93	O22	04	P15	09
18	O04	10	P24	05	56	O17	03	P10	04	94	O23	04	P16	09
19	O05	10	P25	05	57	O18	03	P11	04	95	O24	04	P17	09
20	O06	10	P26	05	58	O19	03	P12	04	96	O02	05	P02	14
21	O07	10	P27	05	59	O20	03	P13	04	97	O03	05	P03	14
22	O08	10	P28	05	60	O21	03	P14	04	98	O04	05	P04	14
23	O09	10	P29	05	61	O22	03	P15	04	99	O05	05	P05	14
24	O17	10	P22	09	62	O23	03	P16	04	100	O06	05	P06	14
25	O18	10	P23	09	63	O24	03	P17	04	101	O07	05	P07	14
26	O19	10	P24	09	64	O02	09	P02	08	102	O08	05	P08	14
27	O20	10	P25	09	65	O03	09	P03	08	103	O09	05	P09	14
28	O21	10	P26	09	66	O04	09	P04	08	104	O17	05	P10	14
29	O22	10	P27	09	67	O05	09	P05	08	105	O18	05	P11	14
30	O23	10	P28	09	68	O06	09	P06	08	106	O19	05	P12	14
31	O24	10	P29	09	69	O07	09	P07	08	107	O20	05	P13	14
32	O02	07	P22	06	70	O08	09	P08	08					
33	O03	07	P23	06	71	O09	09	P09	08					
34	O04	07	P24	06	72	O17	09	P10	08					
35	O05	07	P25	06	73	O18	09	P11	08					
36	O06	07	P26	06	74	O19	09	P12	08					
37	O07	07	P27	06	75	O20	09	P13	08					



ZERO 00	ZERO 01	ZERO 02	NN BIT 0	1	2	3	4	5
0	0	0	0	0	0	0	0	0
1	0	0	NNA0	NNA1	NNA2	NNA3	1	0
X	1	0	NNB0	NNB1	NNB2	NNB3	0	1
X	X	1	NNC0	NNC1	NNC2	NNC3	1	1

NOTES
① FOR RIGHT SHIFT IN RANK 1 SN107 IS USED FOR SIGN EXTENSION

② FOR RIGHT SHIFT IN RANK 2 B107 IS USED FOR SIGN EXTENSION

CONTROL DATA CANADIAN DEVELOPMENT DIVISION

SHIFT NETWORK RANK 1 & 2
NORMALIZE NETWORK

CODE IDENT		DWG NO	REV
34570	D	19981800	D
PUB NO		SHEET CPU3 10	PAGE NO 5-2-23

DETAILED PAK DIAGRAM (CPU 3.11)

SHIFT NETWORK
RANKS 3 & 4

SHIFT NETWORK RANK 3

The third rank of the shift network provides for right and left shifts of 4, 8 and 12. The SK register bits 2 and 3 determine whether the output from rank 2 (SN2 0-107) will be shifted by this third rank. RS determines the shift direction, left or right. The shift network third rank output feeds the fourth rank of the shift network.

SHIFT NETWORK RANK 4

The fourth rank of the shift network provides for right and left shifts of 1, 2 and 3. The SK register bits 0 and 1 determine whether the output from rank 3 (SN3 0-107) will be shifted by this fourth rank. RS determines the shift direction, left or right.

The fourth rank also provides the right shift capabilities for bits 108-113. Compare/move operations, using a maximum shift count of 54, require that the last characters from bit positions 48-53 are gated to bits 108-113. The SK register bits 1 and 2 determine whether the output from rank 1 (S2A 48-53) will be shifted to bits 108-113 by this fourth rank.

SHIFT NETWORK LOGIC LAYOUT

The FH modules perform the right and left shifts for the upper 60 bits (48-107) from the C register. The lower 48 bits (0-47) from C can be right shifted only. This shifting occurs on the FL modules. An FL module also performs the right shift of bits 48-53 to rank 4 bits 108-113.

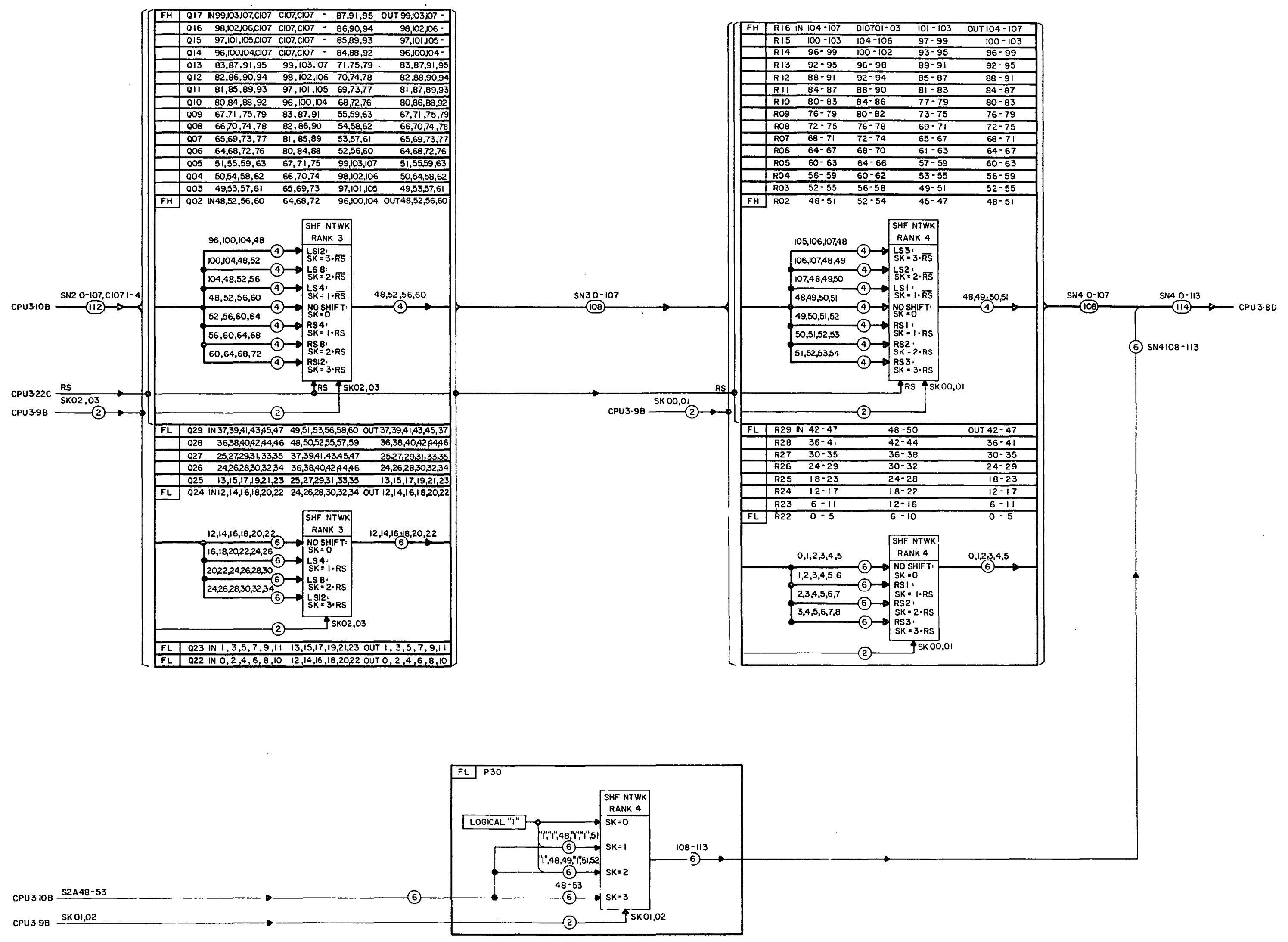
Left shifts are circular with the high order bits starting at 107, reentering at bit 48. Only the upper 60 bits may be left shifted (FH modules). The entire 114 bits may be right shifted. Right shifts are end off with sign extension.

TABLE 5-2-12. CPU 3.11 KEY TEST POINTS

	FH/FL		FH/FL	
BIT NO.	PAK LOC.	SN2 (IN)	PAK LOC.	SN3 (IN)
00	Q22	07	R22	07
01	Q23	07	R22	05
02	Q22		R22	06
03	Q23		R22	
04	Q22	05	R22	09
05	Q23	05	R22	08
06	Q22	09	R23	07
07	Q23	09	R23	05
08	Q22	06	R23	06
09	Q23	06	R23	
10	Q22	08	R23	09
11	Q23	08	R23	08
12	Q24	07	R24	07
13	Q25	07	R24	05
14	Q24		R24	06
15	Q24		R24	
16	Q24	05	R24	09
17	Q25	05	R24	08
18	Q24	09	R25	07
19	Q25	09	R25	05
20	Q24	06	R25	06
21	Q25	06	R25	
22	Q24	08	R25	09
23	Q25	08	R25	08
24	Q26	07	R26	07
25	Q27	07	R26	05
26	Q26		R26	06
27	Q27		R26	
28	Q26	05	R26	09
29	Q27	05	R26	08
30	Q26	09	R27	07
31	Q27	09	R27	05
32	Q26	06	R27	06
33	Q27	06	R27	
34	Q26	08	R27	09
35	Q27	08	R27	08
36	Q28	07	R28	07
37	Q29	07	R28	05

	FH/FL		FH/FL	
BIT NO.	PAK LOC.	SN2 (IN)	PAK LOC.	SN3 (IN)
38	Q28		R28	06
39	Q29		R28	
40	Q28	05	R28	09
41	Q29	05	R28	08
42	Q28	09	R29	07
43	Q29	09	R29	05
44	Q28	06	R29	06
45	Q29	06	R29	
46	Q28	08	R29	09
47	Q29	08	R29	08
48	Q02	04	R02	04
49	Q03	04	R02	08
50	Q04	04	R02	09
51	Q05	04	R02	14
52	Q02	08	R03	04
53	Q03	08	R03	08
54	Q04	08	R03	09
55	Q05	08	R03	14
56	Q02	09	R04	04
57	Q03	09	R04	08
56	Q04	09	R04	09
59	Q05	09	R04	14
60	Q02	14	R05	04
61	Q03	14	R05	08
62	Q04	14	R05	09
63	Q05	14	R05	14
64	Q06	04	R06	04
65	Q07	04	R06	08
66	Q08	04	R06	09
67	Q09	04	R06	14
68	Q06	08	R07	04
69	Q07	08	R07	08
70	Q08	08		
71	Q09	08		
72	Q06	09		
73	Q07	09		
74	Q08	09		
75	Q09	09		

	FH/FL		FH/FL	
BIT NO.	PAK LOC.	SN2 (IN)	PAK LOC.	SN3 (IN)
76	Q06	14	R09	04
77	Q07	14	R09	08
78	Q08	14	R09	09
79	Q09	14	R09	14
80	Q10	04	R10	04
81	Q11	04	R10	08
82	Q12	04	R10	09
83	Q13	04	R10	14
84	Q10	08	R11	04
85	Q11	08	R11	08
86	Q12	08	R11	09
87	Q13	08	R11	14
88	Q10	09	R12	04
89	Q11	09	R12	08
90	Q12	09	R12	09
91	Q13	09	R12	14
92	Q10	14	R13	04
93	Q11	14	R13	08
94	Q12	14	R13	09
95	Q13	14	R13	14
96	Q14	04	R14	04
97	Q15	04	R14	08
98	Q16	04	R14	09
99	Q17	04	R14	14
100	Q14	08	R15	04
101	Q15	08	R15	08
102	Q16	08	R15	09
103	Q17	08	R15	14
104	Q14	09	R16	04
105	Q15	09	R16	08
106	Q16	09	R16	09
107	Q17	09	R16	14



FH	Q17	IN99,103,107,C107	C107,C107 -	87,91,95	OUT 99,103,107 -
	Q16	98,102,106,C107	C107,C107 -	86,90,94	98,102,106 -
	Q15	97,101,105,C107	C107,C107 -	85,89,93	97,101,105 -
	Q14	96,100,104,C107	C107,C107 -	84,88,92	96,100,104 -
	Q13	83,87,91,95	99,103,107	71,75,79	83,87,91,95
	Q12	82,86,90,94	98,102,106	70,74,78	82,88,90,94
	Q11	81,85,89,93	97,101,105	69,73,77	81,87,89,93
	Q10	80,84,88,92	96,100,104	68,72,76	80,86,88,92
	Q09	67,71,75,79	83,87,91	55,59,63	67,71,75,79
	Q08	66,70,74,78	82,86,90	54,58,62	66,70,74,78
	Q07	65,69,73,77	81,85,89	53,57,61	65,69,73,77
	Q06	64,68,72,76	80,84,88	52,56,60	64,68,72,76
	Q05	51,55,59,63	67,71,75	99,103,107	51,55,59,63
	Q04	50,54,58,62	66,70,74	98,102,106	50,54,58,62
	Q03	49,53,57,61	65,69,73	97,101,105	49,53,57,61
FH	Q02	IN48,52,56,60	64,68,72	96,100,104	OUT48,52,56,60

FL	Q29	IN 37,39,41,43,45,47	49,51,53,56,58,60	OUT 37,39,41,43,45,37
	Q28	36,38,40,42,44,46	48,50,52,55,57,59	36,38,40,42,44,46
	Q27	25,27,29,31,33,35	37,39,41,43,45,47	25,27,29,31,33,35
	Q26	24,26,28,30,32,34	36,38,40,42,44,46	24,26,28,30,32,34
	Q25	13,15,17,19,21,23	25,27,29,31,33,35	13,15,17,19,21,23
FL	Q24	IN12,14,16,18,20,22	24,26,28,30,32,34	OUT 12,14,16,18,20,22

FL	Q23	IN 1,3,5,7,9,11	13,15,17,19,21,23	OUT 1,3,5,7,9,11
FL	Q22	IN 0,2,4,6,8,10	12,14,16,18,20,22	OUT 0,2,4,6,8,10

FH	R16	IN 104-107	D10701-03	101-103	OUT 104-107
	R15	100-103	104-106	97-99	100-103
	R14	96-99	100-102	93-95	96-99
	R13	92-95	96-98	89-91	92-95
	R12	88-91	92-94	85-87	88-91
	R11	84-87	88-90	81-83	84-87
	R10	80-83	84-86	77-79	80-83
	R09	76-79	80-82	73-75	76-79
	R08	72-75	76-78	69-71	72-75
	R07	68-71	72-74	65-67	68-71
	R06	64-67	68-70	61-63	64-67
	R05	60-63	64-66	57-59	60-63
	R04	56-59	60-62	53-55	56-59
	R03	52-55	56-58	49-51	52-55
FH	R02	48-51	52-54	45-47	48-51

FL	R29	IN 42-47	48-50	OUT 42-47
	R28	36-41	42-44	36-41
	R27	30-35	36-38	30-35
	R26	24-29	30-32	24-29
	R25	18-23	24-28	18-23
	R24	12-17	18-22	12-17
	R23	6-11	12-16	6-11
FL	R22	0-5	6-10	0-5

CONTROL DATA CANADIAN DEVELOPMENT DIVISION	SHIFT NETWORK RANK 3 & 4	CODE IDENT 34570	D	DWG. NO. 19981800	REV A
		PUB NO.		SHEET CPU3·11	PAGE NO. 5-2-25

CARLETON INSTRUMENTS 138-3 300-12 74

DETAILED PAK DIAGRAM (CPU 3.12)

I3, I3 COMPLEMENT CONTROL

I3 SELECTOR

The I3 selector provides 18-bit input selection to the I3 complement control. Inputs to I3 are received from the following circuits:

Normalize Network NN 0-5	CPU 2.8
I39 0-17	CPU 2.28
K 0-14, U3 0-2	CPU 2.1
A Register 0-17	CPU 2.2
B Register 0-17	CPU 2.2
I0 Selector 0-17	CPU 2.3
X Register 0-17	CPU 2.2
X Register 48-57, $\overline{58}$	CPU 2.2

Selection through I3 is enabled by generation, at the GC modules, of the appropriate select command (e.g., SELBI3 selects B through I3 to the I3 complement control). If no selection is made, zeros are gated through I3.

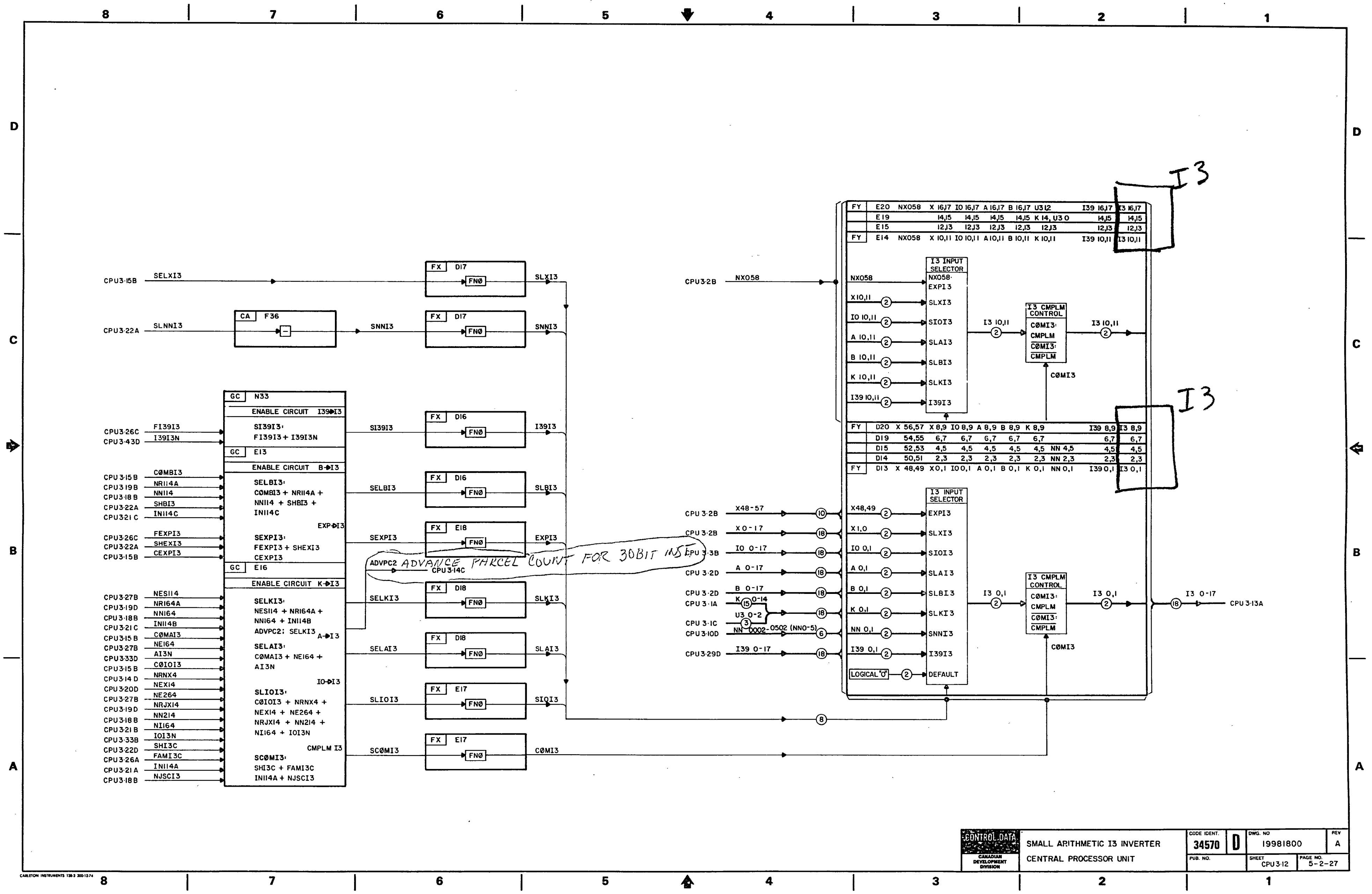
I3 COMPLEMENT CONTROL

The I3 complement control allows the I3 output to be complemented before it is sent to I2 and the E register. COMI3 from the GC module allows I3 to be complemented.

TABLE 5-2-13. CPU 3.12 KEY TEST POINTS

	FY		
BIT NO.	PAK LOC	I0 (IN)	I3
00	D13	14	06
01	D13		04
02	D14	14	06
03	D14		04
04	D15	14	06
05	D15		04
06	D19	14	06
07	D19		04
08	D20	14	06
09	D20		04
10	E14	14	06
11	E14		04
12	E15	14	06
13	E15		04
14	E19	14	06
15	E19		04
16	E20	14	06
17	E20		04

BIT NO.	PAK LOC	X (IN)
48	D13	12
49	D13	11
50	D14	12
51	D14	11
52	D15	12
53	D15	11
54	D19	12
55	D19	11
56	D20	12
57	D20	11



SELXI3
SLNNI3
SNNI3
SLXI3
F39I3
I39I3N
SI39I3
I39I3
ENABLE CIRCUIT I39→I3
SI39I3: FI39I3 + I39I3N
ENABLE CIRCUIT B→I3
SELBI3: CØMBI3 + NRI14A + NNI14 + SHBI3 + INI14C
EXP→I3
SEXPI3: FEXPI3 + SHEXI3 CEXPI3
ENABLE CIRCUIT K→I3
SELKI3: NESI14 + NRI64A + NNI64 + INI14B
ADVPC2: SELKI3
A→I3
SELAI3: CØMAI3 + NEI64 + AI3N
IO→I3
SLIOI3: CØIOI3 + NRNX4 + NEXI4 + NE264 + NRJXI4 + NN2I4 + NII64 + IOI3N
CMPLM I3
SCØMI3: SHI3C + FAMI3C INI14A + NJSCI3
ADVPC2 ADVANCE PARCEL COUNT FOR 30BIT INST
CPU3-14C
I3 INPUT SELECTOR
I3 CMPLM CONTROL
CØMI3: CMPLM
CØMI3: CMPLM
I3 10,11
I3 0,1
I3 0-17
CPU3-13A
LOGICAL "0"
DEFAULT
I3
I3

SMALL ARITHMETIC I3 INVERTER

CENTRAL PROCESSOR UNIT

CODE IDENT.		DWG. NO	REV
34570	D	19981800	A
PUB. NO.		SHEET CPU3-12	PAGE NO. 5-2-27

DETAILED PAK DIAGRAM (CPU 3.13)

F REGISTER, E REGISTER
I2, F ADDER

F REGISTER

The F register and E register together serve as input feeders to the F adder. The F register also functions as the output register for results from the F adder. Input to F is from I2. The output of F feeds the F adder and the F register fanouts which distribute the results computed in the F adder.

I2 SELECTOR

The I2 selector provides 18-bit selection of data from I3 or the F adder to the F register. I3I2 controls selection of I3 through I2 to F. The absence of I3I2 allows the F adder to be selected through I2 to F. The generation of I3I2 from GC module F12 also generates ENABF from the same GC module. This allows I3 data to be selected through I2 and automatically gated into F.

F ADDER

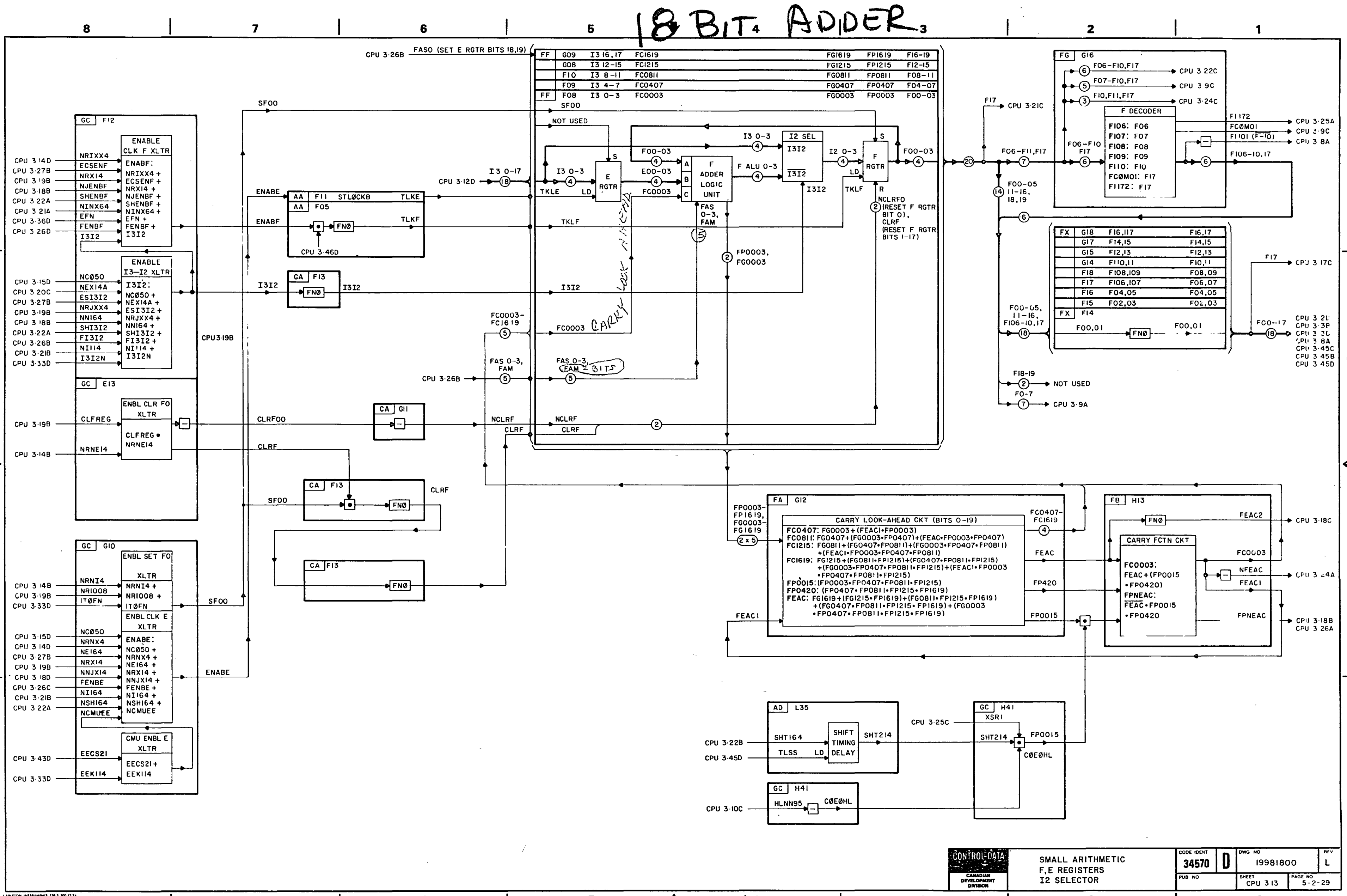
The F adder consists of a high speed arithmetic logic unit (ALU) capable of performing both arithmetic and logical functions. Arithmetic logic operations are selected by the FAS 0-3, FAM signals. Group carry propagate (FP0003 - FP1619) and carry generate (FG0003 - FG1619) signals from the F-ALU are sent to the first stage carry look-ahead control on FA module G12. The first stage carry look-ahead provides internal carry signals (FC0407 - FC1619) back to the F-ALU. The first stage carry look-ahead also provides group propagate (FP0015, FP0420) and end around carry (FEAC) signals to the second stage carry function control on FB module H13. The carry function control provides the end around internal carry signal (FC0003) back to the F-ALU, and distributes end around carry (FEAC) to the processor controls.

In its normal operation, the F adder performs a ones complement full add operation for: increment instructions (CPU 3.21), normal jump, return jump instructions (CPU 3.18, 3.19), compare/move address sequence calculations (CPU 3.33-3.37); also P register advancement, addition of RA to the absolute memory address for initial start, and full RNI operations. In addition, the F adder performs various functions for exponent manipulation or ones counter for FAD/FMD instructions (CPU 3.24, 3.25, 3.26).

E REGISTER

The E register and the F register together serve as input feeders to the F adder. The E register receives its input from I3. Its output feeds the F adder.

18 BIT ADDER



GC F12
ENABLE CLK F XLTR
ENABF: NRIXX4 + ECSENF + NRXI4 + NJENBF + SHENBF + NINX64 + EFN + FENBF + I3I2
ENABLE I3–I2 XLTR
I3I2: NCØ50 + NEXI4A + ESI3I2 + NRJXX4 + NNI64 + SHI3I2 + FI3I2 + NII14 + I3I2N
GC E13
ENBL CLR FO XLTR
CLFREG • NRNEI4
GC G10
ENBL SET FO XLTR
NRNI4 + NRI008 + ITØFN
ENBL CLK E XLTR
ENABE: NCØ50 + NRNX4 + NEI64 + NRXI4 + NNJXI4 + FENBE + NII64 + NSHI64 + NCMUEE
CMU ENBL E XLTR
EECS2I + EEKI14
AA F11 STLØCKB TLKE
AA F05 TLKF
CA F13 I3I2
CA G11
CA F13 CLRF
CA F13
FASO (SET E RGTR BITS 18,19)
E RGTR
F ADDER LOGIC UNIT
I2 SEL
F RGTR
NCLRFO (RESET F RGTR BIT 0), CLRF (RESET F RGTR BITS 1–17)
CARRY LOOK AHEAD
FAS 0-3, FAM 2 BITS
FG G16
F DECODER
FI06: F06
FI07: F07
FI08: F08
FI09: F09
FI10: FI0
FCØMOI: FI7
FII72: FI7
FB H13
CARRY FCTN CKT
FC0003: FEAC+(FP0015 •FP0420)
FPNEAC: FEAC•FP0015 •FP0420
AD L35 SHIFT TIMING DELAY
GC H41 XSRI SHT214 FP0015 CØEØHL
GC H41 HLNN95 CØEØHL

FF	G09	I3 16,17	FC1619	FG1619	FP1619	F16–19
	G08	I3 12–15	FC1215	FG1215	FP1215	F12–15
	F10	I3 8–11	FC0811	FG0811	FP0811	F08–11
	F09	I3 4–7	FC0407	FG0407	FP0407	F04–07
FF	F08	I3 0–3	FC0003	FG0003	FP0003	F00–03

FX	G18	F16,117	F16,17
	G17	F14,15	F14,15
	G15	F12,13	F12,13
	G14	F110,11	F10,11
	F18	F108,109	F08,09
	F17	F106,107	F06,07
	F16	F04,05	F04,05
	F15	F02,03	F02,03
FX	F14	F00,01	F00,01

FA G12

CARRY LOOK-AHEAD CKT (BITS 0–19)

FC0407: FG0003+(FEACI•FP0003)
FC0811: FG0407+(FG0003•FP0407)+(FEAC•FP0003•FP0407)
FC1215: FG0811+(FG0407•FP0811)+(FG0003•FP0407•FP0811)
+(FEACI•FP0003•FP0407•FP0811)
FC1619: FG1215+(FG0811•FP1215)+(FG0407•FP0811•FP1215)
+(FG0003•FP0407•FP0811•FP1215)+(FEACI•FP0003
•FP0407•FP0811•FP1215)
FP0015: (FP0003•FP0407•FP0811•FP1215)
FP0420: (FP0407•FP0811•FP1215•FP1619)
FEAC: FG1619+(FG1215•FP1619)+(FG0811•FP1215•FP1619)
+(FG0407•FP0811•FP1215•FP1619)+(FG0003
•FP0407•FP0811•FP1215•FP1619)

CONTROL DATA CANADIAN DEVELOPMENT DIVISION

SMALL ARITHMETIC
F,E REGISTERS
I2 SELECTOR

CODE IDENT		DWG NO	REV
34570	D	19981800	L
PUB NO		SHEET CPU 3 13	PAGE NO 5-2-29

CARLETON INSTRUMENTS 138 3 300-12 74

DETAILED PAK DIAGRAM (CPU 3.14)

RNI SEQUENCE

The RNI sequence controls the operations necessary to initially start the CPU following an exchange jump, and performs RNIs for subsequent instructions to be executed. RNIs are referred to as an initial start RNI, a full RNI, or a parcel RNI. An initial start RNI is enabled by the following conditions:

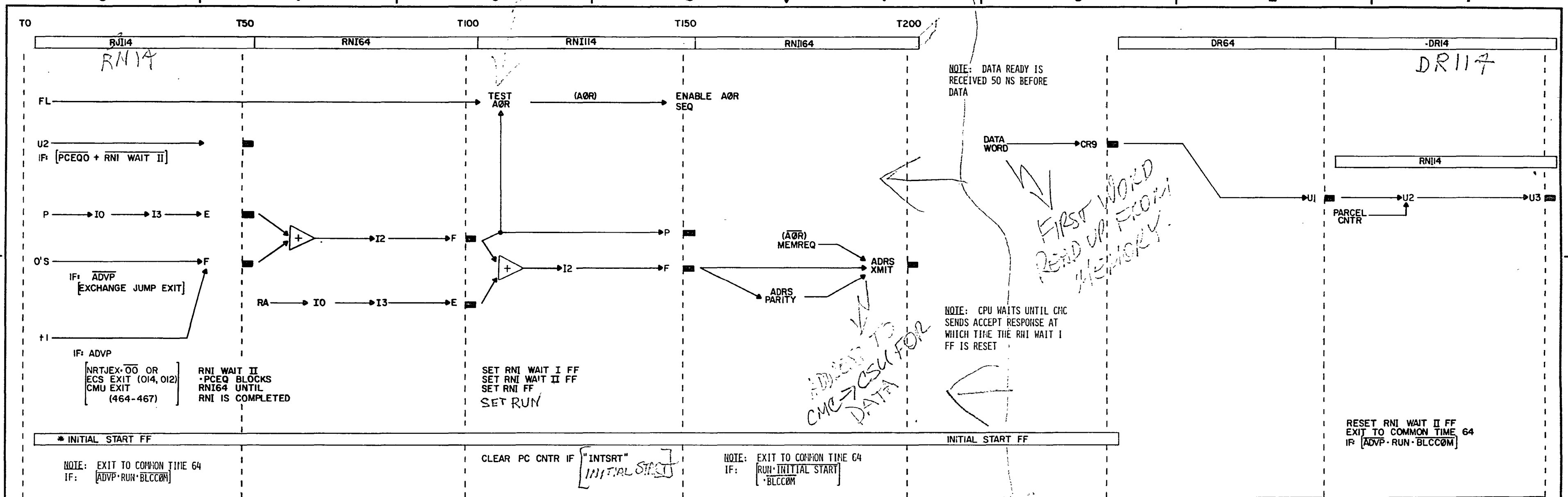
1.	EXJONC . 2EX814	Exchange Jump Exit
2.	NESETR	Normal ECS Exit
3.	NJPEX1	Normal Jump Condition Met
4.	NRTJEX	Return Jump Exit
5.	NCMUEX	CMU Exit

An initial start RNI obtains a new 60-bit instruction word from central memory. A full RNI is similar in operation to an initial start RNI in that a new 60-bit instruction word is obtained from central memory; however, a full RNI is initiated between execution of the first and second instructions of the word being processed. A parcel RNI obtains the next 15-bit parcel for execution within a 60-bit instruction word. A parcel RNI is initiated when a sequence exit (SEQEXT) is present along with the condition of $\overline{\text{ADVP}}$. This indicates that an additional parcel is available for execution within the current instruction word.

PARCEL COUNTER

The parcel counter located on the GL module is a 2-bit counter that is incremented sequentially through counts of 0, 1, 2 and 3. The counts correspond to the four parcels of each central memory instruction word, and provide gating controls to selector U2. The counter is advanced during each RNI sequence to enable the entrance of the next 15-bit parcel into the instruction translation network. If a translated parcel is found to be part of a 30-bit instruction, the parcel counter will be incremented to ensure that K is not taken as the next instruction.

ADDRESS OUT OF RANGE



NOTE: DATA READY IS RECEIVED 50 NS BEFORE DATA

NOTE: CPU WAITS UNTIL CMC SENDS ACCEPT RESPONSE AT WHICH TIME THE RNI WAIT I FF IS RESET

[NRTJEX·OO OR ECS EXIT (014, 012) CMU EXIT (464-467)]

RNI WAIT II ·PCEQ BLOCKS RNI64 UNTIL RNI IS COMPLETED

SET RNI WAIT I FF
SET RNI WAIT II FF
SET RNI FF
SET RUN

RESET RNI WAIT II FF
EXIT TO COMMON TIME 64
IF: [$\overline{\text{ADVP}}$·RUN·$\overline{\text{BLCCØM}}$]

* INITIAL START FF

INITIAL START FF

NOTE: EXIT TO COMMON TIME 64
IF: [$\overline{\text{ADVP}}$·RUN·$\overline{\text{BLCCØM}}$]

CLEAR PC CNTR IF ["INTSRT" INITIAL START]

NOTE: EXIT TO COMMON TIME 64
IF: [RUN·$\overline{\text{INITIAL START}}$ ·$\overline{\text{BLCCØM}}$]

TERM NAME	COMMAND OR FUNCTION	DPD NO
RNI-T14		14
NRNE4A	P → I0	14
SI00		3
NRNE4A		14
NRNIX4		14
SLI0I3	I0 → I3	12
ENABE	ENABLE E RGTR	13
RNI-T14	[ADVP]	14
NRNI4		14
SCLRF	CLR F	13
SF00	+1 → F	13
RNI-T14		
NRNE14	[RNI EXJ FIF]	14
NCLRF	CLR F	13
CLRF00	0 → F0	13
RNI-T14		14
RNI14	[$\overline{\text{ADVP}}$]	14
RNIEXT	[RUN·$\overline{\text{BLCCOM}}$] (RNI EXIT TO COMMON TIME)	14
RNI·T0	[$\overline{\text{PCEQ}}$ + $\overline{\text{RNTII}}$]	14
U2U3	ENABLE U3 RGTR	14

TERM NAME	COMMAND OR FUNCTION	DPD NO
$\overline{\text{SI00}}$.$\overline{\text{ST01}}$.$\overline{\text{SI02}}$	RA → I0	3
RNI-T64		14
NRNIX4		14
SLI0I3	I0 → I3	12
ENABE	ENABLE E RGTR	12
RNI-I64		14
NRIXX4		14
ENABF	ENABLE F RGTR	13

TERM NAME	COMMAND OR FUNCTION	DPD NO
RNI-T114		
NRIXX4		14
ENABF	ENABLE F RGTR	13
RNI-T114		
RI114		
ENABØR	TEST AØR	17
ENBP	ENABLE P RGTR	3
RNI-T114	[INTSRT]	
CLERPC	CLEAR PARCEL COUNTER	14

TERM NAME	COMMAND OR FUNCTION	DPD NO
RNI-T164		
NRI164	[$\overline{\text{AØR}}$]	14
CEMREQ		17
MEMREQ	MEMREQ → ADRS XMIT	17
EMBADT	CLOCK ADRS XMIT	17
RNI-T114		
RNIEXT	[RUN·$\overline{\text{NINSRT}}$ ·$\overline{\text{BLCCØM}}$]	14

* NOTE:
INITIAL START FF SET BY:
1) CMU EXIT "NCMUEX"
2) EXCHANGE JUMP EXIT "2EX814·EXJONC"
3) ECS EXIT "NESETR"
4) NORMAL JUMP EXIT CONDITION MET "NJPEX1"
5) RETURN JUMP EXIT FROM 00·CEJ ENABLED OR 010

TERM NAME	COMMAND OR FUNCTION	DPD NO
DR64		16
FORCEX	[INTSRT·RIWTII]	14
SEQEXT	ENABLE SEQ EXIT	14
DR64		
ENU1	ENABLE U1 RGTR	16

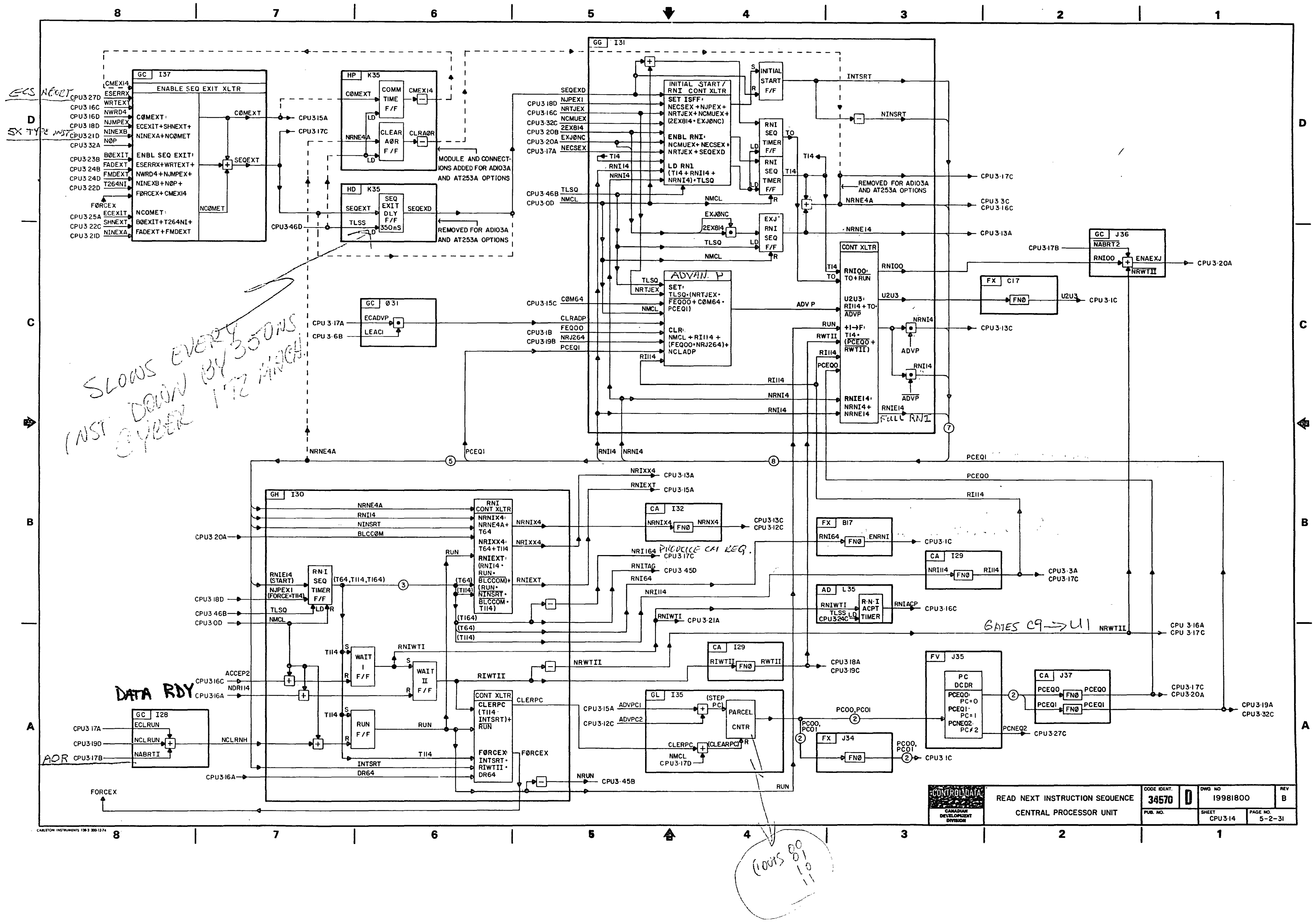
TERM NAME	COMMAND OR FUNCTION	DPD NO
NDR114		16
RIWTII	RESET RNI WAIT II FF	14
RNI-T14		14
U2U3	ENABLE U3 RGTR	14
RNI-T14		14
RNI14	[$\overline{\text{ADVP}}$]	14
RNIEXT	[RUN·$\overline{\text{BLCCOM}}$] (RNI EXIT TO COMMON TIME)	14

P+RA IS XMITTED AS A MEMORY REQUEST & ARE WAITING FOR DATA READY

CONTROL DATA CANADIAN DEVELOPMENT DIVISION	INSTRUCTION FLOW INITIAL START RNI	CODE IDENT. 34570 D	DWG. NO. 19981800 REV A
		FIGURE 5-2-1	PAGE NO. 5-2-30·2

1. EXCH JUMP EXIT — SETS INITIAL START AND EXCHANGE EXIT
2. EXCH EXIT GENERATES RNIE14 (FULL RNI)
3. RNI T64 → RNI 114 CLEAR ADVANCE P, SET RNI WAIT 1 AND 2, SET RUN, CLEAR P.C.
4. RNI 164 – REQUEST MEMORY
5. ACCEPT CLEARS WAIT 1
6. DATA READY GENERATES FORCEX
7. FORCEX GENERATES SEQEXT WHICH CLEARS INITIAL START
8. RNI T00 AND T14 U2 → U3 ($\overline{\text{RNI WAIT} \cdot \text{PC=0}} = \overline{\text{RNI WAIT}} + \overline{\text{PC=0}}$)
9. PARCEL RNI GENERATES RNI EXT WHICH SETS COM T50, T64
10. COM T50 ADVANCE P.C. PC = 1 }
11. COM T64 SET ADVANCE P } START SEQUENCE
12. SEQUENCE EXIT RNI T00, T14 WITH ADVANCE P SET = FULL RNI
13. CLEAR ADVANCE P, REQUEST MEMORY, SET COM T50, T64, SET WAIT I, II
14. SEQUENCE EXIT, RNI T00, 14, PARCEL RNI ($\overline{\text{PC=0}}$ GATE THAT MAKES TO $\boxed{\text{ADV P SET?}}$)

RNI SEQ.



CONTROL DATA CANADIAN DEVELOPMENT DIVISION

READ NEXT INSTRUCTION SEQUENCE
CENTRAL PROCESSOR UNIT

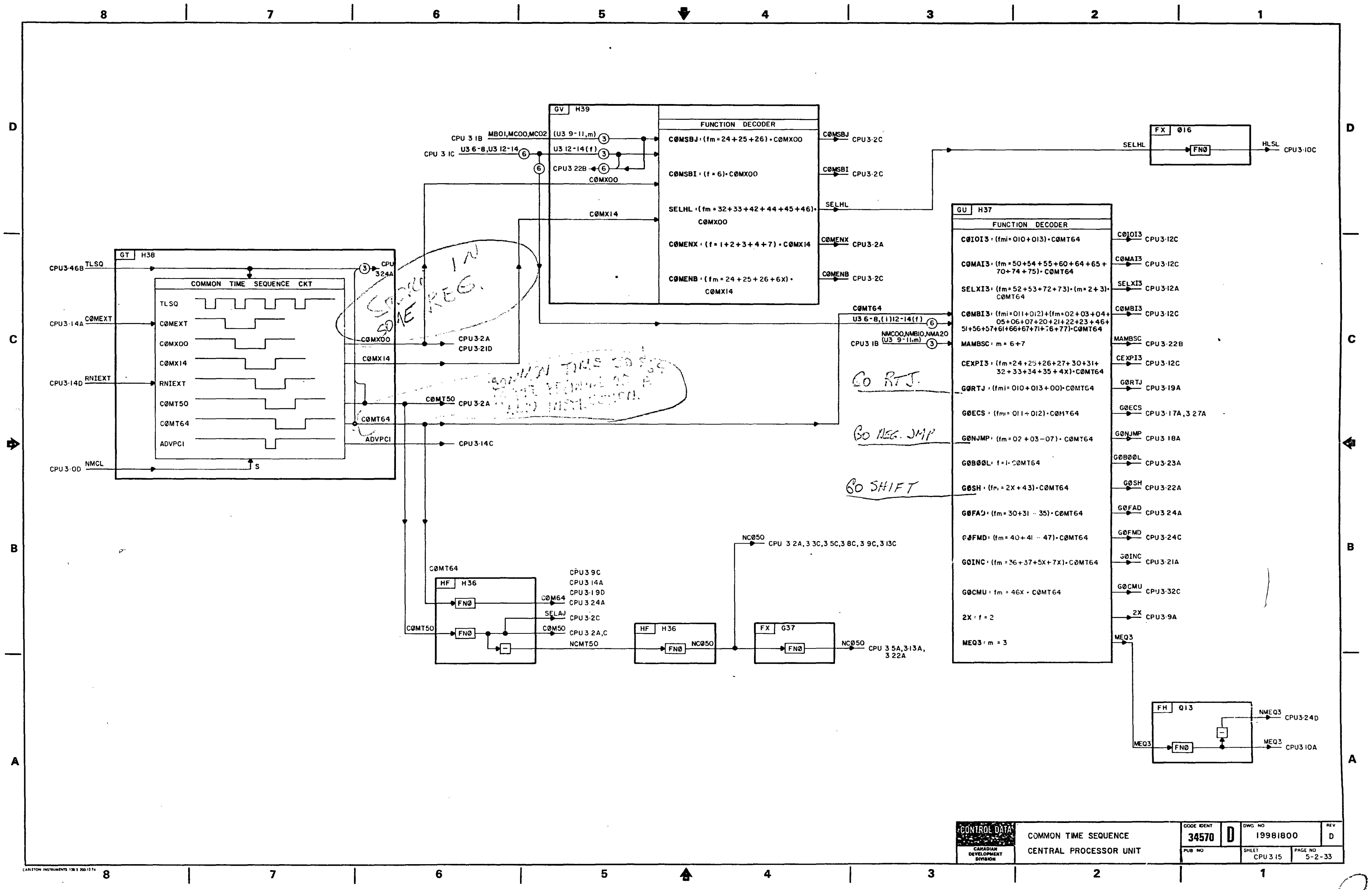
CODE IDENT.		DWG NO	REV
34570	D	19981800	B
PUB. NO.		SHEET CPU3-14	PAGE NO. 5-2-31

DETAILED PAK DIAGRAM (CPU 3.15)

COMMON TIME SEQUENCE

The common time sequence controls the initial operating conditions of each of the instruction sequences. Common time 50/64 is initiated by RNIEXT, and provides instruction translation and decoding operations to initiate the appropriate control sequence for execution. Common time 0/14 is initiated by COMEXT, and provides controls to return the results of an instruction to a selected B or X register. [illegible]

The common time sequence timing chain is contained on the GT module. At common time 50, ADVPC1 is generated to update the parcel counter pointing to the next 15-bit instruction. Common time 64 (COMT64) enables the function decode translator circuits located on the GU module. The GU module provides a decode of the instruction in U3, and generates the appropriate GO signal to the control sequence for the instruction type decoded.


GT H38
COMMON TIME SEQUENCE CKT
TLSQ
CØMEXT
CØMXOO
CØMX14
RNIEXT
CØMT50
CØMT64
ADVPCI
CPU3-46B TLSQ
CPU3-14A CØMEXT
CPU3-14D RNIEXT
CPU3-0D NMCL
S
CPU 324A
CPU3-2A
CPU3-21D
CPU3-14C
GV H39
FUNCTION DECODER
CPU 3 1B MB01,MC00,MC02 (U3 9-11,m)
CPU 3 1C U3 6-8,U3 12-14 (f)
CPU3 22B
CØMSBJ = (fm = 24+25+26) · CØMXOO
CØMSBI = (f = 6) · CØMXOO
SELHL = (fm = 32+33+42+44+45+46) · CØMXOO
CØMENX = (f = 1+2+3+4+7) · CØMX14
CØMENB = (fm = 24+25+26+6X) · CØMX14
CPU3-2C
CPU3-2A
FX Ø16
SELHL
FNØ
HLSL CPU3-10C
GU H37
FUNCTION DECODER
CØIOI3 = (fm = 010+013) · CØMT64
CØMAI3 = (fm = 50+54+55+60+64+65+70+74+75) · CØMT64
SELXI3 = (fm = 52+53+72+73) · (m = 2+3) · CØMT64
CØMBI3 = (fm = 011+012) + (fm = 02+03+04+05+06+07+20+21+22+23+46+51+56+57+61+66+67+71+76+77) · CØMT64
MAMBSC = m = 6+7
CEXPI3 = (fm = 24+25+26+27+30+31+32+33+34+35+4X) · CØMT64
GØRTJ = (fm = 010+013+00) · CØMT64
GØECS = (fm = 011+012) · CØMT64
GØNJMP = (fm = 02+03-07) · CØMT64
GØBØØL = f = 1 · CØMT64
GØSH = (fm = 2X+43) · CØMT64
GØFAD = (fm = 30+31 ··· 35) · CØMT64
GØFMD = (fm = 40+41 ··· 47) · CØMT64
GØINC = (fm = 36+37+5X+7X) · CØMT64
GØCMU = fm = 46X · CØMT64
2X = f = 2
MEQ3 = m = 3
U3 6-8,(i)12-14(f)
NMC00,NMB10,NMA20 (U3 9-11,m)
CPU3 1B
CPU3-12C
CPU3-12A
CPU3-22B
CPU3-19A
CPU3-17A, 3 27A
CPU3 18A
CPU3-23A
CPU3-22A
CPU3 24A
CPU3-24C
CPU3-21A
CPU3-32C
CPU3-9A
FH Q13
NMEQ3 CPU3-24D
MEQ3 CPU3-10A
HF H36
CØMT64
CØMT50
CØM64 CPU3 9C, CPU3 14A, CPU3-19D, CPU3 24A
SELAJ CPU3-2C
CØM50 CPU3 2A,C
NCMT50
HF H36 NCØ50
FX G37
NCØ50 CPU 3 2A, 3 3C, 3 5C, 3 8C, 3 9C, 3 13C
NCØ50 CPU 3 5A, 3 13A, 3 22A
Go RTJ.
Go NEG. JMP
Go SHIFT
SEND IN SOME REG.
[illegible]

CARLETON INSTRUMENTS

CONTROL DATA CANADIAN DEVELOPMENT DIVISION	COMMON TIME SEQUENCE CENTRAL PROCESSOR UNIT	CODE IDENT 34570	D	DWG NO 19981800	REV D
		PUB NO		SHEET CPU 3 15	PAGE NO 5-2-33

GOOD SINK POINTS
GO SIGNALS

DETAILED PAK DIAGRAM (CPU 3.16)

ACCEPT SEQUENCE

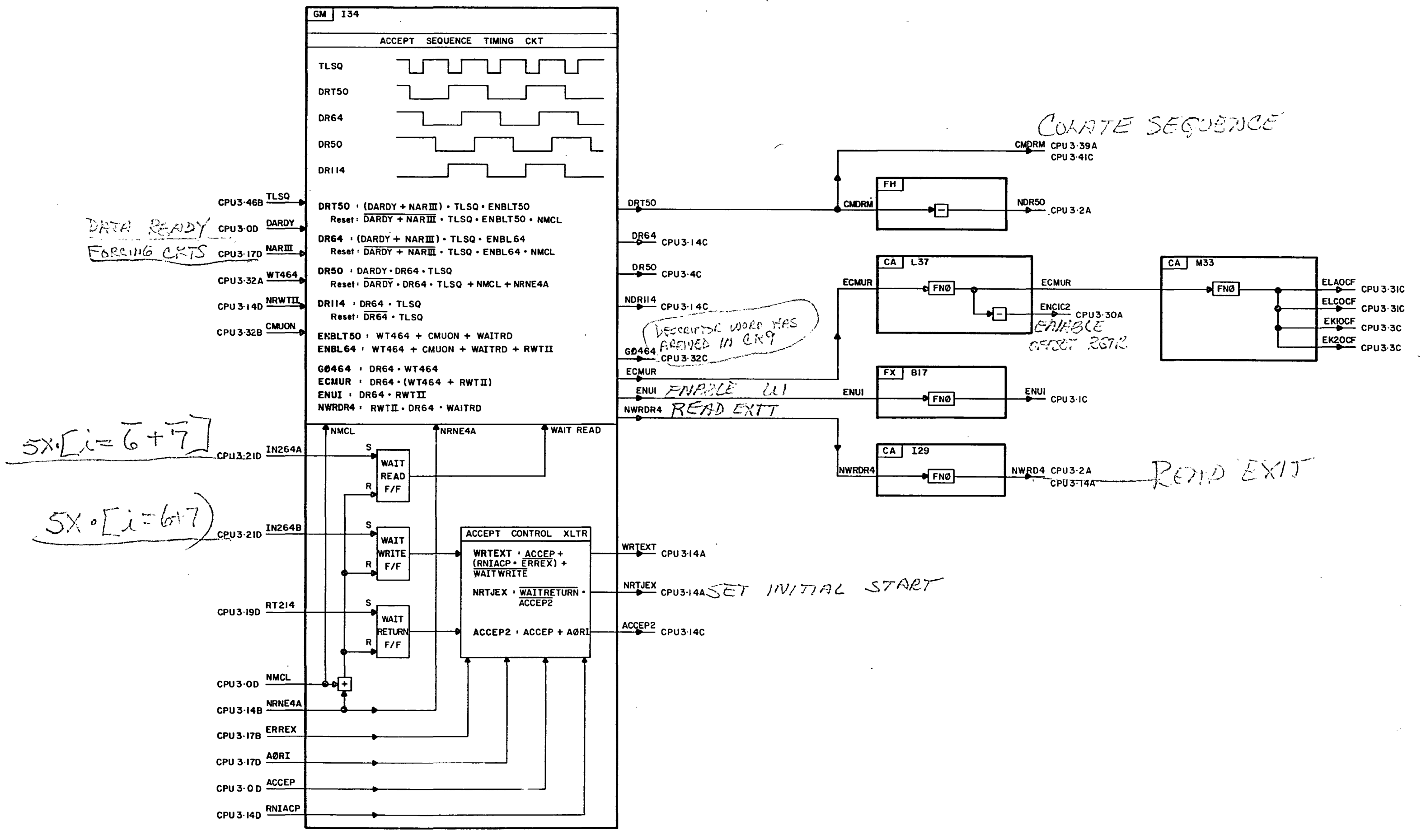
The accept sequence consists of a timing chain and control that are enabled by CMC data ready (DARDY). Data ready is sent 50 ns ahead of output data from CMC. The accept sequence generates control signals that enable the acceptance of data.

NDR50 selects operand register Xi for receipt of a data operand as the result of an increment read memory reference.

DR64 enables an RNI initial start sequence exit (FORCEX) after receipt of a new 60-bit instruction word.

CMDRM is used by the compare/move control. It indicates receipt of a data word to allow start of the data sequence.

When an address out of range fault occurs, NARIII from the AOR sequence forces the start of a false accept sequence. Setting of the CLR CR9 FF (CPU 3.17) during AOR ensure that zeros are gated to the CR9 register.



CONTROL DATA CANADIAN DEVELOPMENT DIVISION	ACCEPT SEQUENCE	CODE IDENT. 34570	D	DWG NO. 19981800	REV A
		PUB. NO.		SHEET CPU3-16	PAGE NO 5-2-35

DETAILED PAK DIAGRAM (CPU 3.17)

AOR SEQUENCE

The AOR sequence timing located on the GL module is enabled when an address out of range condition is detected. Address out of range may be detected as a result of one of the following conditions:

1. Increment read address out of range
2. Increment write address out of range
3. RNI or branch out of range
4. Compare/move instruction with:
   - (a) C1 or C2 > 9, or
   - (b) K1 or K2 address out of range
5. ECS address out of range check.

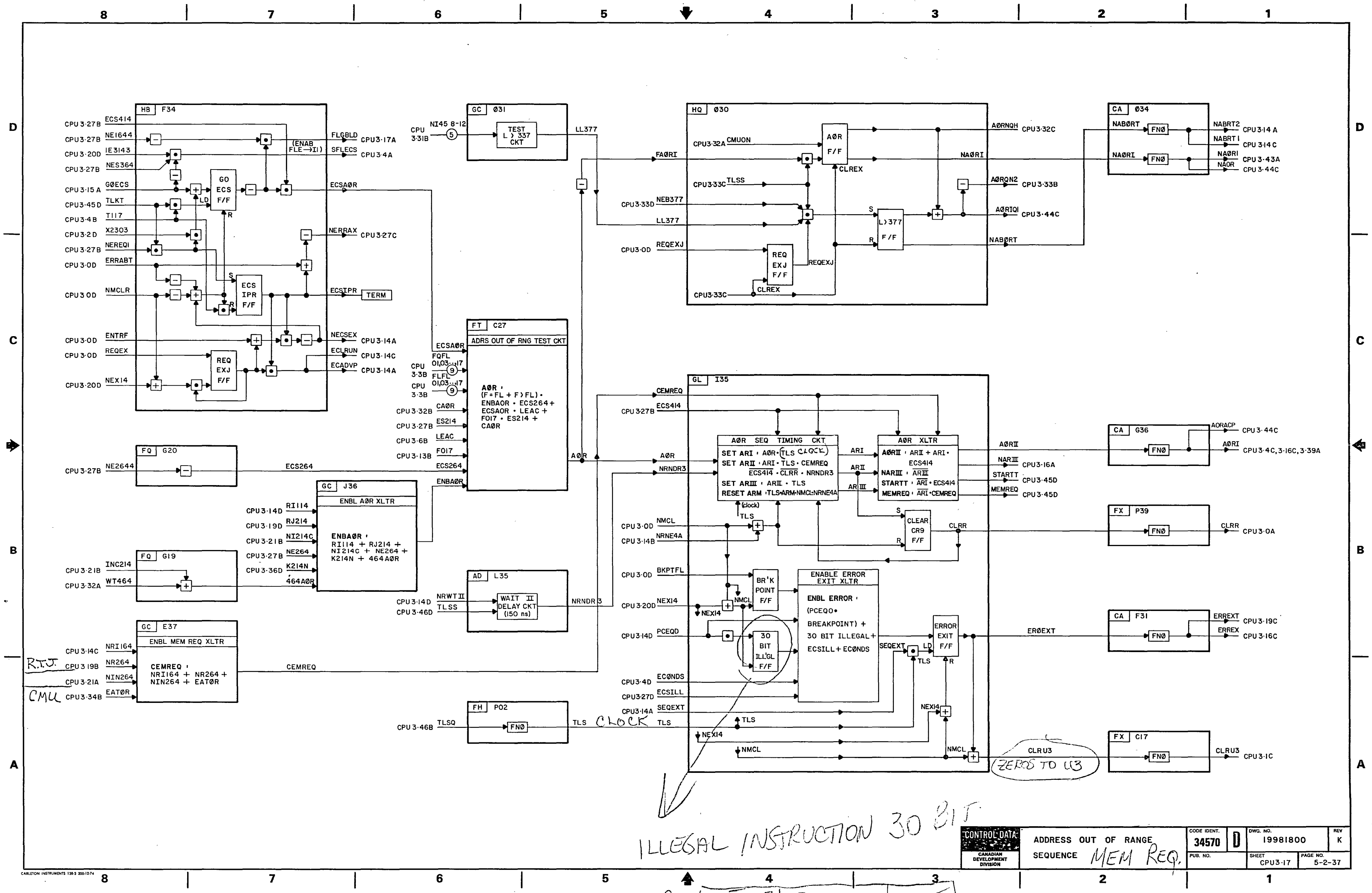
The AOR sequence aborts the out of range memory request, the compare/move, or ECS instruction and sets the clear CR9 FF. This allows for the transfer of zeros from CR9 to the selected X register when an address out of range occurs on an increment read instruction. Blocking the memory request also prevents a write operation on an increment write instruction. Address out of range sets bit 48 of the error exit register (CPU 3.4). If the corresponding exit mode register selection was also made, exit condition sensed (ECONDS) enables setting the error exit FF at sequence exit time.

ERROR EXIT RESPONSES

The error exit FF, also located on the GL module, is set at sequence exit time when one of the following error conditions is detected:

1. Exit condition sensed (ECONDS)
   - (a) Bit 48 Address out of range
   - (b) Bit 49 Infinite condition
   - (c) Bit 50 Indefinite condition
   - (d) Bit 51 Flag operation parity error condition
   - (e) Bit 52 CMC input parity error condition
   - (f) Bit 53 CM data parity error condition
2. Illegal instruction
3. Breakpoint sensed
4. 30-bit Illegal FF.

Setting of the error exit FF clears the U3 instruction register and enables a return jump error exit sequence (CPU 3.19).



il | 15 BITS | 30 BITS | 15 BITS

CONTROL DATA
CANADIAN DEVELOPMENT DIVISION

ADDRESS OUT OF RANGE SEQUENCE MEM REQ.

CODE IDENT.		DWG. NO.	REV
34570	D	19981800	K
PUB. NO.		SHEET CPU3-17	PAGE NO. 5-2-37

CARLETON INSTRUMENTS 138-3 200-12-74

DETAILED PAK DIAGRAM (CPU 3.18)

NORMAL JUMP SEQUENCE

The normal jump sequence controls the operations necessary to perform the following instructions:

02ixk	Jump to (Bi) + K
030jk	Branch to K if (Xj) = 0
031jk	Branch to K if (Xj) ≠ 0
032jk	Branch to K if (Xj) Positive
033jk	Branch to K if (Xj) Negative
034jk	Branch to K if (Xj) in Range
035jk	Branch to K if (Xj) Out of Range
036jk	Branch to K if (Xj) Definite
037jk	Branch to K if (Xj) Indefinite
04ijk	Branch to K if (Bi) = (Bj)
05ijk	Branch to K if (Bi) ≠ (Bj)
06ijk	Branch to K if (Bi) ≥ (Bj)
07ijk	Branch to K if (Bi) < (Bj)

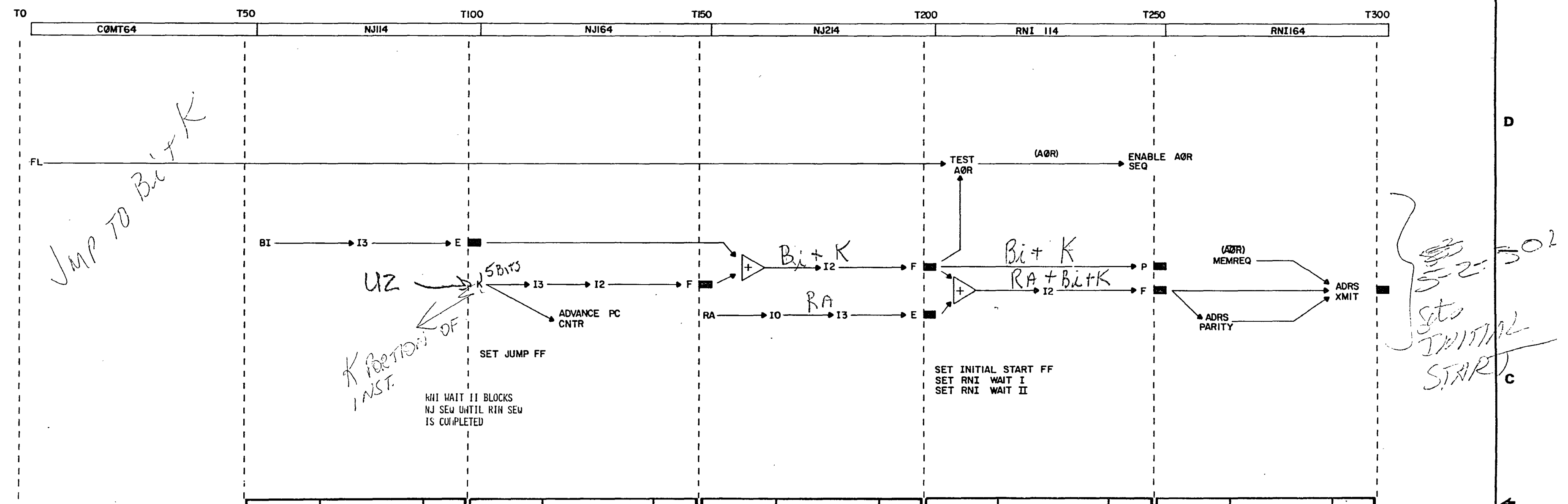
The 02 instruction performs an unconditional jump to the address specified by the contents of index register Bi, plus the K portion of the instruction. The branch address is K when i = 0.

The conditional jump instructions 030-037, 04-07 branch to address K if the jump condition specified by the instruction is met. Jump conditions are defined as follows:

02	Unconditional
030	Xj = All "1's" or "0's" (60 bits)
031	Xj ≠ All "1's" or "0's" (60 bits)
032	$\overline{\text{XSR1}}$ (Xj Positive)
033	XSR1 (Xj Negative)
034	Xj exp ≠ 3777 + 4000 (Xj in Range)
035	Xj exp = 3777 + 4000 (Xj Out of Range)
036	Xj exp ≠ 1777 + 6000 (Xj Definite)
037	Xj exp = 1777 + 6000 (Xj Indefinite)
04	Bi = Bj All Pass . $\overline{\text{EAC}}$
05	Bi ≠ Bj $\overline{\text{All Pass}}$ + EAC
06	Bi ≥ Bj BSR1 . $\overline{\text{BSR2}}$ + BSR1 = BSR2 . F NEGATIVE
07	Bi < Bj $\overline{\text{BSR1}}$. BSR2 + BSR1 = BSR2 . F POSITIVE

A successful test of a specified jump condition effects an RNI initial start operation. The next instruction is executed at address K after RA has been added. An unsuccessful test of the jump condition causes a sequence exit which effects an RNI for the next instruction in the current program.

A jump to an address out of range enables the AOR sequence (CPU 3.17). The CPU response is dependent on whether the appropriate exit mode selection was made and the monitor flag /MEJ/CEJ condition.



TERM NAME	COMMAND OR FUNCTION	DPD NO
NNJ100		18
SELBI	SLECT BI RGTR	2
NNJ114		18
SELBI3	B → I3	12
NNJX14		18
ENABE	ENABLE E RGTR	13

TERM NAME	COMMAND OR FUNCTION	DPD NO
NN164		18
ADVPC2	ADVANCE PC CNTR	12
SELKI3	K → I3	12
NN164		18
I3I2	I3 → I2	13
ENABF	ENABLE F RGTR	

TERM NAME	COMMAND OR FUNCTION	DPD NO
NJ214		
NNJX14		18
ENABE	ENABLE E RGTR	13
$\overline{ST00}\cdot\overline{ST01}$		
$\cdot\overline{ST02}$	RA → I0	3
NN214		18
SLI0I3	I0 → I3	12
NJ214		
NJENBF	ENABLE F RGTR	18
ENABF		13
NJ214	[JUMP FF]	18
NJPEX1	ENABLE RNI T114, SET INITIAL START FF	18

TERM NAME	COMMAND OR FUNCTION	DPD NO
RNI·Y114		
NRIXX4		14
ENABF	ENABLE F RGTR	13
RI114		14
ENABØR	TEST AØR	17
ENBP	ENABLE P RGTR	3

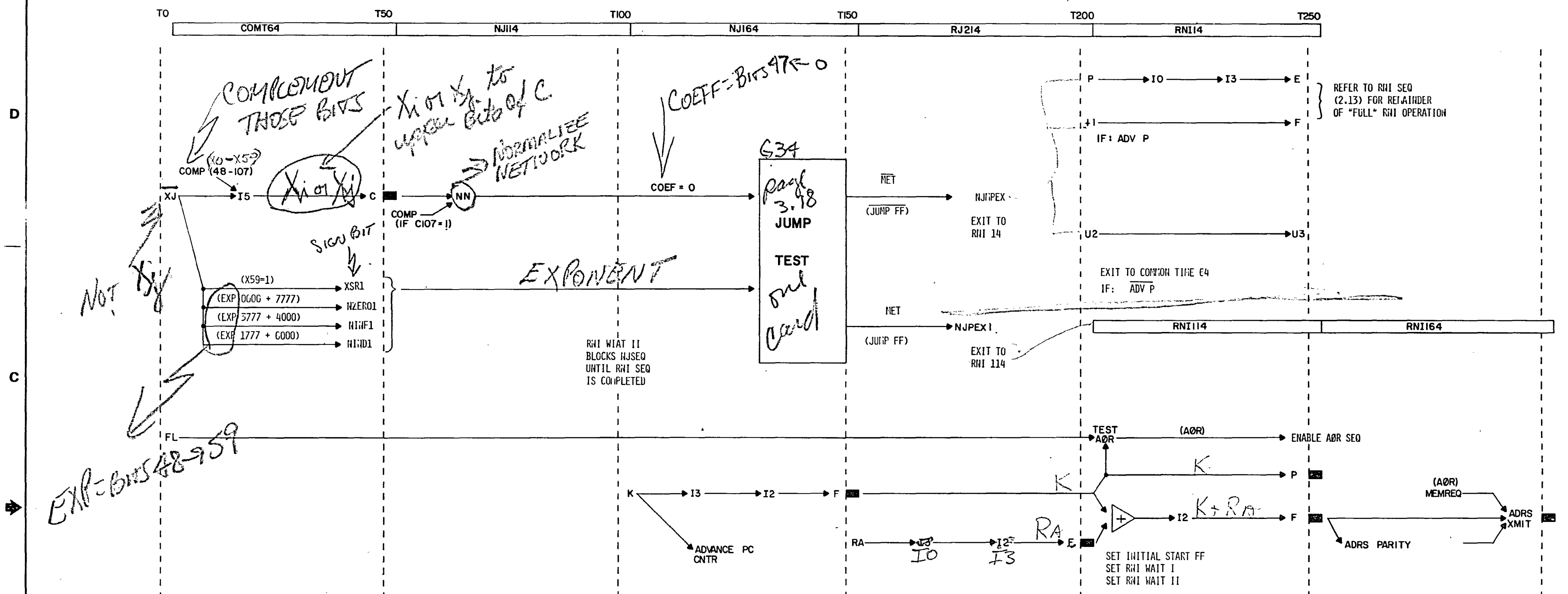
TERM NAME	COMMAND OR FUNCTION	DPD NO
NRI164		14
CEMREQ		17
ENBADT	ENABLE ADRS XMIT	45
MEMREQ	MEMREQ → ADRS XMIT	17

CONTROL DATA
CANADIAN DEVELOPMENT DIVISION

INSTRUCTION FLOW
SEQUENCE: NORMAL JUMP
INSTRUCTION: 02

CODE IDENT		DWG NO	REV
34570	D	19981800	A
FIGURE 5-2-2		PAGE NO 5-2-38.2	

CARLETON INSTRUMENTS 136-3 200-12-74



TERM NAME	COMMAND OR FUNCTION	DPD NO
CØM50		15
SELXJ	SELECT XJ RGTR	2
NCØ64		15
I5185		7
I5285	$X \longrightarrow I5$	7
I5164	0-59 48-107	7
I5264	(I594·I552·$\overline{I551}$)	7
I5236		7
I5436		7
CØM64		15
I5C85	COMP I5 48-95	8
I5C67	COMP I5 96-107	8
CØN64		15
XSR1	SET XSR1 (IF X59=1)	24
CØM64		
NINF1	SET NINF1 (IF EXP=3777+4000)	24
NIND1	SET NIND1 (IF EXP=1777+6000)	24
NZERO1	SET NZERO1 (IF EXP=0000+7777)	24

TERM NAME	COMMAND OR FUNCTION	DPD NO
NN114	ENABLE NORMALITE	18
ADVPC2	NETWORK	10

TERM NAME	COMMAND OR FUNCTION	DPD NO
NN164		18
ADVPC2	ADVANCE PC CNTR	12
SELKI3	$K \longrightarrow I3$	12
NN164		18
I3I2	$I3 \longrightarrow I2$	13
ENABF	ENABLE F RGTR	

TERM NAME	COMMAND OR FUNCTION	DPD NO
NN214		
NNJX14		18
ENABE	ENABLE E RGTR	13
SI00·SI01		
·SI02	$RA \longrightarrow I0$	3
NN214		18
NJENBF		18
ENABF	ENABLE F RGTR	13
NJ214	[$\overline{\text{JUMP FF}}$]	18
NJMPEX	ENABLE RNI T14	18
NJ214	[JUMP FF]	18
NNPEX1	ENABLE RNI T114, SET INITIAL START FF	18

TERM NAME	COMMAND OR FUNCTION	DPD NO
RNI-T14		14
NRNE4A	$P \longrightarrow I0$	14
SI00		3
RNI-T14		
NRNIX4		14
SLI0I3	$I0 \longrightarrow I3$	12
ENABE	ENABLE E RGTR	13
NRNI4		14
SCLRF	[ADV P]	13
SF00	$+1 \longrightarrow F$	13
RNI14	[$\overline{\text{ADV P}}$]	14
RNIEXT	ENABLE COMMON TIME 64	14
RNI-T64		
NRIXX4		14
ENABF	ENABLE F RGTR	13
RI114		14
ENABØR	TEST AØR	17
ENBP	ENABLE P RGTR	3

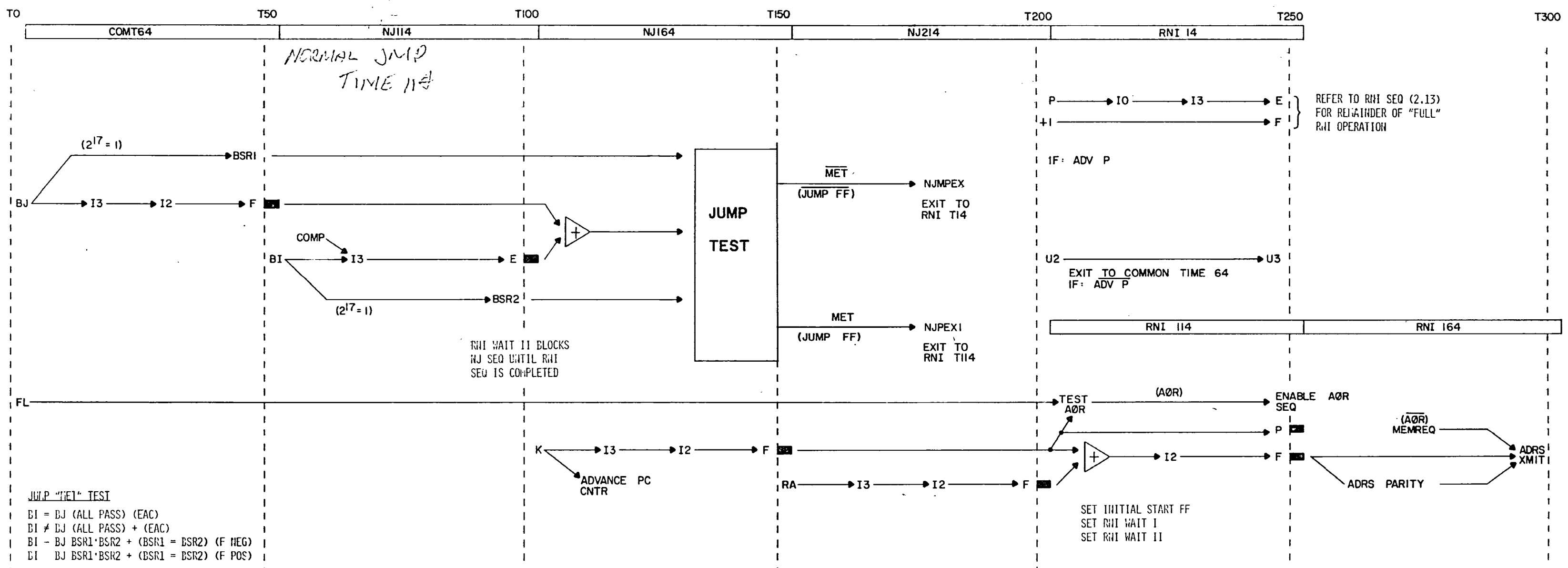
TERM NAME	COMMAND OR FUNCTION	DPD NO
NRI164		14
CEMREQ		17
ENBADT	ENABLE ADRS XMIT	45
MEMREQ	$\text{MEMREQ} \longrightarrow \text{ADRS XMIT}$	17

CONTROL DATA CANADIAN DEVELOPMENT DIVISION

INSTRUCTION FLOW
SEQUENCE : NORMAL JUMP
INSTRUCTION : 030 - 037

CODE IDENT 34570 D | DWG. NO. 19981800 | REV A

FIGURE 5-2-3 | PAGE NO. 5-2-38-3



TERM NAME	COMMAND OR FUNCTION	DPD NO
CØM50		15
SELBJ	SELECT BJ RGTR	2
CØMT64		15
COMBI3	B ⟶ I3	12
NCØ64		15
I3I2	I3 ⟶ I2	13
ENABF	ENABLE F RGTR	13
CØM64		14
BSR1	SET BSR1 (1F B17=1)	24

TERM NAME	COMMAND OR FUNCTION	DPD NO
NNJ100		18
SELBI	SELECT BI RGTR	2
NN114		18
SELBI3	B ⟶ I3	12
NJ114		18
NNJX14		18
ENABE	ENABLE E RGTR	13
NN114		18
ENXSR2		24
BSR2	SET BSR2 (1F B17=1)	24

TERM NAME	COMMAND OR FUNCTION	DPD NO
NN164		18
ADVPC2	ADVANCE PC CNTR	12
SELKI3	K ⟶ I3	12
NN164		18
I3I2	I3 ⟶ I2	13
ENABF	ENABLE F RGTR	

TERM NAME	COMMAND OR FUNCTION	DPD NO
NJ214		
NNJX14		18
ENABE	ENABLE E RGTR	13
$\overline{\text{SI00}}\cdot\overline{\text{SI01}}$		
$\cdot\overline{\text{SI02}}$	RA ⟶ I0	3
NN214		18
NJENBF		18
ENABF	ENABLE F RGTR	13
NJ214	[$\overline{\text{JUMP FF}}$]	18
NJMPEX	ENABLE RNI T14	18
NJ214		18
NJPEX1	[JUMP FF] (ENABLE RNI T114 SET INITIAL START FF)	18

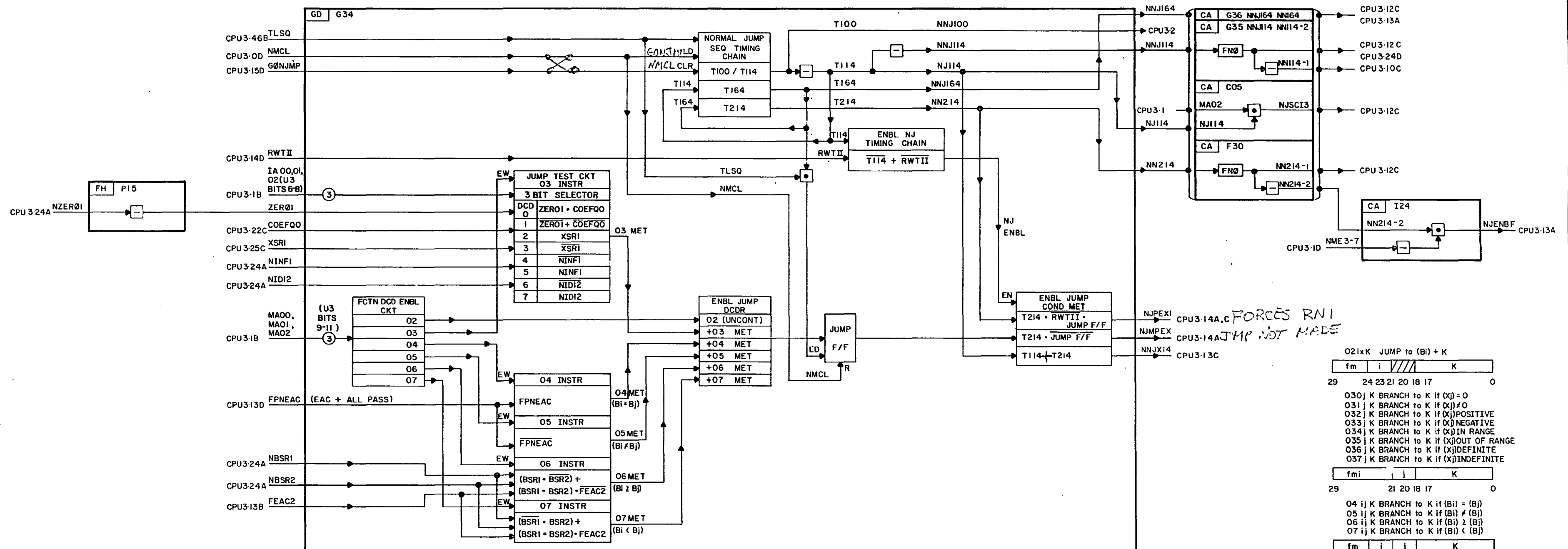
TERM NAME	COMMAND OR FUNCTION	DPD NO
RNI-T14		14
NRNE4A		14
SI00	P ⟶ I0	3
RNI-T14		
NRNIX4		14
SLIOI3	I0 ⟶ I3	12
ENABE	ENABLE E RGTR	13
NRNI4	[ADV P]	
SCLRF		
SF00	+1 ⟶ F	
RNI14	[$\overline{\text{ADV P}}$]	14
RNIEXT	ENABLE COMMON TIME 64	14
RNI-T114		
NRIXX4		14
ENABF	ENABLE F RGTR	13
RI114		14
ENHBØR	TEST AØR	17
ENBP	ENABLE P RGTR	14

TERM NAME	COMMAND OR FUNCTION	DPD NO
NRI164		14
CEMREQ		17
ENBHDT	ENABLE HDRS XMIT	45
MEMREQ	MEMREQ ⟶ ADRS XMIT	17

CONTROL DATA CANADIAN DEVELOPMENT DIVISION

INSTRUCTION FLOW
SEQUENCE: NORMAL JUMP
INSTRUCTION: 04-07

CODE IDENT.		DWG NO.	REV
34570	D	19981800	A
FIGURE 5-2-4		PAGE NO 5-2-38-4	



021xK JUMP to (Bi) + K

fm	i	//////	K
29 … 24	23 … 21	20 … 18	17 … 0

030j K BRANCH to K if (Xj) = 0
031 j K BRANCH to K if (Xj) ≠ 0
032 j K BRANCH to K if (Xj) POSITIVE
033 j K BRANCH to K if (Xj) NEGATIVE
034 j K BRANCH to K if (Xj) IN RANGE
035 j K BRANCH to K if (Xj) OUT OF RANGE
036 j K BRANCH to K if (Xj) DEFINITE
037 j K BRANCH to K if (Xj) INDEFINITE

fmi	j	K
29 … 21	20 … 18	17 … 0

04 ij K BRANCH to K if (Bi) = (Bj)
05 ij K BRANCH to K if (Bi) ≠ (Bj)
06 ij K BRANCH to K if (Bi) ≥ (Bj)
07 ij K BRANCH to K if (Bi) < (Bj)

fm	i	j	K
29 … 24	23 … 21	20 … 18	17 … 0

CONTROL DATA CANADIAN DEVELOPMENT DIVISION	NORMAL JUMP SEQUENCE	CODE IDENT 34570	D	DWG NO 19981800	REV A
		PUB NO		SHEET CPU3·18	PAGE NO 5-2-39

CARLETON INSTRUMENTS 138-3 200-12-74

DETAILED PAK DIAGRAM (CPU 3.19)

RETURN JUMP SEQUENCE

The return jump sequence controls operations necessary to perform the following instructions:

00xxx	Monitor Stop (Error Exit to MA)
010xK	Return Jump to K
013jK	Central Exchange Jump

RETURN JUMP 010

The 010 instruction stores an unconditional jump instruction (0400) to the current program address plus one (P + 1) in the upper half of memory location K + RA, then branches to K + 1 + RA for the next instruction.

A jump to an address out of range enables the AOR sequence (CPU 3.17). The CPU response is dependent on whether the appropriate exit mode selection was made and the monitor flag /MEJ/CEJ condition.

CENTRAL EXCHANGE JUMP 013

The 013 instruction is enabled or disabled by the MEJ/CEJ switch on the dead start panel. If the switch is enabled, the return jump sequence allows the processor to send an exchange request (EXJREQ) to CMC. CMC then responds with request exchange (REQEXJ), which enables the exchange jump sequence (CPU 3.20) and unconditionally exchange jumps the CPU, regardless of the state of the monitor flag bit. However, instruction action differs depending on whether the monitor flag bit is set or clear:

Monitor Flag clear

The exchange sequence (CPU 3.20) makes the starting address for the exchange from the 18-bit monitor address register (MA). During the exchange, the monitor flag bit is set.

Monitor Flag set

The exchange sequence (CPU 3.20) takes the 18-bit starting address formed by adding K to the contents of Bj during the return jump sequence. During the exchange, the monitor flag bit is cleared.

If the MEJ/CEJ switch is in the disable position, the 013 instruction is illegal. The return jump sequence detects the illegal condition. NSETIL sets the illegal FF (CPU 3.27). The illegal FF, in turn, sets the error exit FF (CPU 3.17). Error exit clears the U3 instruction register and forces a return jump error exit sequence.

MONITOR STOP (Error Exit to MA) 00

The 00 instruction is enabled or disabled by the MEJ/CEJ switch on the dead start panel. The return jump sequence is enabled by an instruction decode of 00, or by an error exit that caused the U3 register to be cleared and thus forced an instruction decode of 00. With the MEJ/CEJ switch in the disable position, the processor has no central.exchange or monitor exchange jump capability, so the return jump sequence clears the run FF and stops the processor.

In the enable position, the processor has the exchange jump capability, so the 00 instruction is executed in two passes through the return jump sequence.

The first pass records at RA a monitor stop instruction (00), the exit condition bits (EE), and the program address at exit time in the following format:

Store at RA

$$00 \quad \underbrace{00}_{\text{Error bits}} \quad \underbrace{xxxxxx}_{\text{P Register (Program Address)}} \quad 0000000000$$

The return jump sequence then waits until CMC responds with an accept. Since the return jump wait FF is set, accept allows setting the RNI initial start FF by generating NRTJEX (CPU 3.16). The RNI sequence, using the P register cleared to zeros during the first return jump pass, reads the 00 instruction at RA. The 00 instruction enables the return jump sequence for the second pass. The second pass generates exchange request (EXJREQ) if the monitor flag is clear. If the monitor flag is set, the run FF is cleared and the CPU stops.

The error response conditions with MEJ/CEJ enabled and monitor flag set or clear, or MEJ/CEJ disabled, are detailed tables 5-2-14 and 5-2-15.

MONITOR FLAG

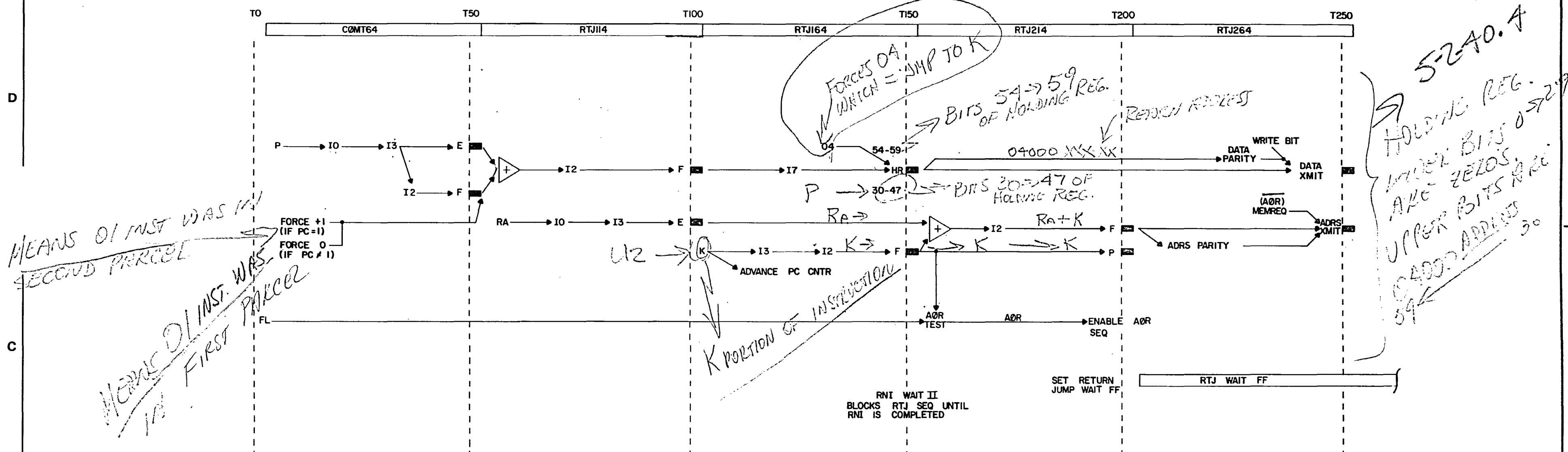
TABLE 5-2-14. ERROR RESPONSE WITH MEJ/CEJ ENABLED, MF SET

Error Condition	Error Response: Exit Mode Selected	Error Response: Exit Mode Not Selected
Illegal instruction	1. Execute the illegal instruction as if it were a pass. 2. Stop CPU. 3. Store P and exit condition bits at RAC. 4. Clear P.	1. Execute the illegal instruction as if it were a pass. 2. Stop CPU. 3. Store P and exit condition bits at RAC. 4. Clear P.
Exit condition bit 48 set by an increment read of an address out of range	1. Read all zeros to the selected X register. 2. Stop CPU. 3. Store P and exit condition bits at RAC. 4. Clear P.	1. Read all zeros to the selected X register. 2. Continue execution.
Exit condition bit 48 set by an increment write of an address out of range	1. Block write operation, contents of CM is unchanged. 2. Stop CPU. 3. Store P and exit condition bits at RAC. 4. Clear P.	1. Block write operation, contents of CM is unchanged. 2. Continue execution.
Exit condition bit 48 set on RNI or branch out of range	1. Stop CPU. 2. Store P and exit condition bits at RAC. 3. Clear P.	1. Stop CPU. 2. Store P and exit condition bits at RAC. 3. Clear P.
Exit condition bit 48 set on CMU instruction a. C1 or C2 > 9 b. K1 or K2 address out of range	1. Condition (a) causes instruction to execute as pass. Condition (b) causes instruction moves or compares up to the point of address out of range. 2. Stop CPU. 3. Store P and exit condition at RAC. 4. Clear P.	1. Condition (a) causes instruction to execute as pass. Condition (b) causes instruction moves or compares up to the point of address out of range. 2. Continue with next 60-bit instruction.
Exit condition bit 48 set by an ECS address range check	1. Force ECS instruction to execute as a pass instruction. 2. Stop CPU. 3. Store P and exit condition bits at RAC. 4. Clear P.	1. Force ECS instruction to execute as a pass instruction. 2. Exit to next 60-bit word. 3. Continue execution with next 60-bit word.
Infinite condition (bit 49) Indefinite condition (bit 50) **ECS flag register parity (bit 51)** **CMC input error condition (bit 52)** **CM data error condition (bit 53)**	1. Stop CPU. 2. Store P and exit condition bits at RAC. 3. Clear P.	1. Continue execution.
CMC input error condition (bit 52)	1. Block write operation, contents of CM is unchanged. 2. Stop CPU. 3. Store P and exit condition bits at RAC. 4. Clear P.	1. Block write operation, contents of CM is unchanged. 2. Continue execution.
00 instruction	1. Stop CPU. 2. Store P and exit condition bits at RAC. 3. Clear P.	1. Stop CPU. 2. Store P and exit condition bits at RAC. 3. Clear P.
Breakpoint signal from CMC (refer to breakpoint notes)	1. Execute remaining parcels of 60-bit word currently executing. 2. Stop CPU. 3. Store P and exit condition bits at RAC. 4. Clear P.	1. Execute remaining parcels of 60-bit word currently executing. 2. Stop CPU. 3. Store P and exit condition bits at RAC. 4. Clear P.

MONITOR FLAG.

TABLE 5-2-15. ERROR RESPONSE WITH MEJ/CEJ ENABLED, MF CLEAR

Error Condition	Error Response Exit Mode Selected	 Exit Mode Not Selected
Illegal instruction	1. Execute the illegal instruction as if it were a pass. 2. Stop CPU. 3. Store P and exit condition bits at RAC. 4. Clear P. 5. Exchange jump to MA and set MF.	1. Execute the illegal instruction as if it were a pass. 2. Stop CPU. 3. Store P and exit condition bits at RAC. 4. Clear P. 5. Exchange jump to MA and set MF.
Exit condition bit 48 set by an increment read of an address out of range	1. Read all zeros to the selected X register. 2. Stop CPU. 3. Store P and exit condition bits at RAC. 4. Clear P. 5. Exchange jump to MA and set MF.	1. Read all zeros to the selected X register. 2. Continue execution.
Exit condition bit 48 set due to an increment write of an address out of range	1. Block write operation, contents of CM is unchanged. 2. Stop CPU. 3. Store P and exit condition bits at RAC. 4. Clear P. 5. Exchange jump to MA and set MF.	1. Block write operation, contents of CM is unchanged. 2. Continue execution.
Exit condition bit 48 set due to an RNI or branch address out of range	1. Stop CPU. 2. Store P and exit condition bits at RAC. 3. Clear P. 4. Exchange jump to MA and set MF.	1. Stop CPU. 2. Store P and exit condition bits at RAC. 3. Clear P. 4. Exchange jump to MA and set MF.
Exit condition bit 48 set on CMU instruction a. C1 or C2 > 9 b. K1 or K2 address out of range	1. Condition (a) causes instruction to execute as pass. Condition (b) causes instruction moves or compares up to the point of address out of range. 2. Stop CPU. 3. Store P and exit condition at RAC. 4. Clear P. 5. Exchange jump to MA and set MF.	1. Condition (a) causes instruction to execute as pass. Condition (b) causes instruction moves or compares up to the point of address out of range. 2. Continue with next 60-bit instruction.
Exit condition bit 48 set by an ECS address range check	1. Force ECS instruction to execute as a pass instruction. 2. Stop CPU. 3. Store P and exit condition bits at RAC. 4. Clear P. 5. Exchange jump to MA and set MF.	1. Force ECS instruction to execute as a pass instruction. 2. Continue execution with next 60-bit word.
Infinite condition (bit 49) Indefinite condition (bit 50) ECS flag register parity (bit 51) CMC input error condition (bit 52) CM data error condition (bit 53)	1. Stop CPU. 2. Store P and exit condition bits at RAC. 3. Clear P. 4. Exchange jump to MA and set MF.	1. Continue execution.
CMC input error condition (bit 52)	1. Block write operation, contents of CM is unchanged. 2. Stop CPU. 3. Store P and exit condition bits at RAC. 4. Clear P. 5. Exchange jump to MA and set MF.	1. Block write operation, contents of CM is unchanged. 2. Continue execution.
00 instruction	1. Stop CPU. 2. Store P and exit condition bits at RAC. 3. Clear P. 4. Exchange jump to MA and set MF.	1. Stop CPU. 2. Store P and exit condition bits at RAC. 3. Clear P. 4. Exchange jump to MA and set MF.
Breakpoint signal from CMC (refer to breakpoint notes)	1. Execute remaining parcels of 60-bit word currently executing. 2. Stop CPU. 3. Store P and exit condition bits at RAC. 4. Clear P. 5. Exchange jump to MA and set MF.	1. Execute remaining parcels of 60-bit word currently executing. 2. Stop CPU. 3. Store P and exit condition bits at RAC. 4. Clear P. 5. Exchange jump to MA and set MF.



TERM NAME	COMMAND OR FUNCTION	DPD NO
NCØ64		15
S100	P → I0	3
ENABE	ENABLE E RGTR	13
CØ1013		15
SL1013	I0 → I3	12
NCØ64		15
I3I2	I3 → I2	13
EN BF	ENABLE F RGTR	13

TERM NAME	COMMAND OR FUNCTION	DPD NO
NR100B	[PC=1]	19
SCLRF	CLR F RGTR	13
SF00	+1 → F	13
NR100A	[PC≠1]	19
CLRF	CLR F RGTR	13
CLRF00	0 → F	13
S100·S101		
·S102	RA → I0	3
RTJ114		
NRJX14	I0 → I3	19
SL1013		12
RTJ114		
NRX14A		19
ENABE	ENABLE E RGTR	13
ENABF	ENABLE F RGTR	13

TERM NAME	COMMAND OR FUNCTION	DPD NO
RTJ164		19
NR164A		19
SELKI3	K → I3	12
ADVPC2	ADVANCE PC CNTR	13
RTJ164		19
NRJXX4		19
I3I2	I3 → I2	13
ENABF	ENABLE F RGTR	13
RTJ150		
SFI7	F0-17 → I7 30-47	19
ENBHR	ENABLE HR RGTR	45
RTJ164		
NR164A		19
S0417	$04_8 \rightarrow HR_{54-59}$	19

TERM NAME	COMMAND OR FUNCTION	DPD NO
RTJ214		
NRTXX4		19
ENBP	ENABLE P RGTR	3
RJ214		19
ENABØR	TEST AØR	17
RT214		19
RTJWAIT	SET RTJ WAIT FF	16
RTJ214		19
NRX14A	ENABLE F RGTR	19

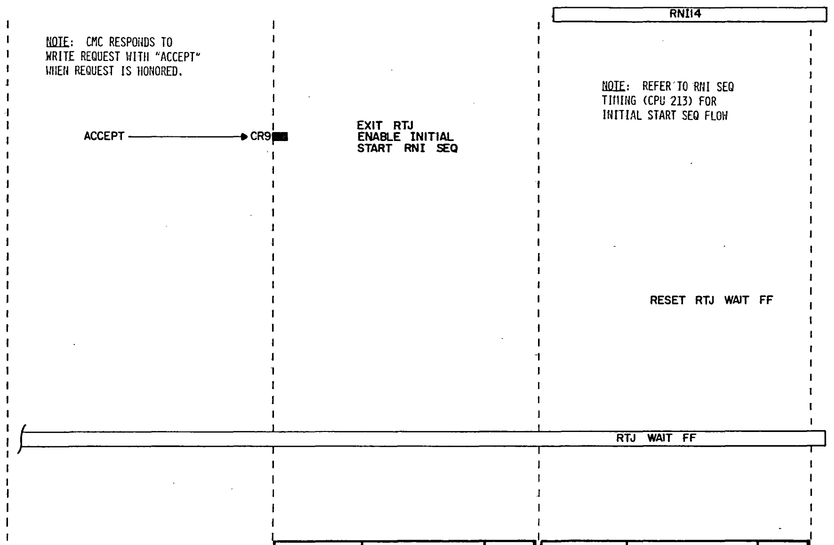
TERM NAME	COMMAND OR FUNCTION	DPD NO
NRJ264		19
CEMREQ	[$\overline{AØR}$]	17
MEMREQ	MEMREQ → ADRS XMIT	17
ENBADT	CLOCK ADRS XMIT	17
NR264		19
ENBWRT	WRITE BIT →	45
DWRITE	DATA XMIT	45
ENBDT	CLOCK DATA XMIT	45

(Part 1 of 2)

CONTROL DATA CANADIAN DEVELOPMENT DIVISION

INSTRUCTION FLOW
SEQUENCE : RETURN JUMP
INSTRUCTION : 010

CODE IDENT.		DWG. NO.	REV
34570	D	19981800	A
FIGURE 5-2-5		PAGE NO. 5-2-40.3	


NOTE: CMC RESPONDS TO WRITE REQUEST WITH "ACCEPT" WHEN REQUEST IS HONORED.
ACCEPT
CR9
EXIT RTJ ENABLE INITIAL START RNI SEQ
RNI14
NOTE: REFER TO RNI SEQ TIMING (CPU 213) FOR INITIAL START SEQ FLOW
RESET RTJ WAIT FF
RTJ WAIT FF

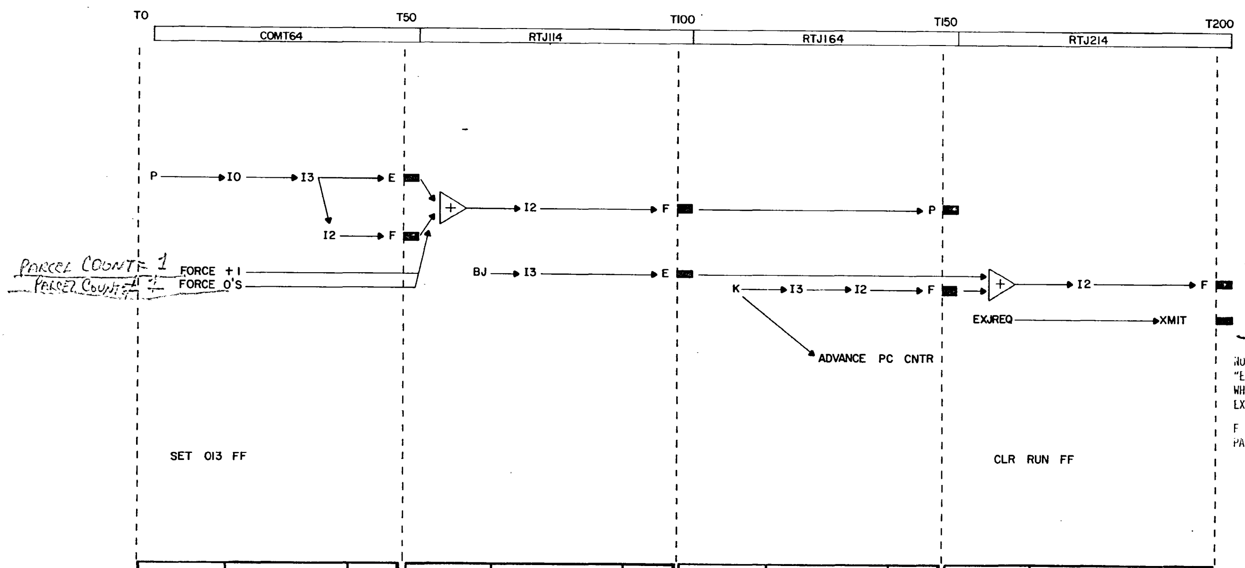
TERM NAME	COMMAND OR FUNCTION	DPD NO
ACCEP2 NRTJEX	RTJ WAIT FF ENABLE INITIAL START RNI SEQ	16

TERM NAME	COMMAND OR FUNCTION	DPD NO
RNI14 NRNE4A	CLR RTJ WAIT FF	16

(Part 2 of 2)

CONTROL DATA CANADIAN DEVELOPMENT DIVISION	INSTRUCTION FLOW SEQUENCE • RETURN JUMP INSTRUCTION : 010	CODE IDENT. 34570	D	DWG. NO. 19981800
		FIGURE 5 - 2 - 5		PAGE NO. 5-2-40-4
				REV A

CARLETON INSTRUMENTS 138-3 200-12-74



} Go To 5-2-42.1

NOTE: CMC RESPONDS TO "EXJREQ" WITH "REQEXJ" WHICH INTURN ENABLES EXCANGE SEQ ~~(2,19)~~ 3.20

F RGTR CONTAINS EXCHANGE PACKAGE ADDRESS

TERM NAME	COMMAND OR FUNCTION	DPD NO
NC064		15
SI00	$P \longrightarrow I0$	3
ENABE	ENABLE E RGTR	13
C01013		15
SLI013	$I0 \longrightarrow 03$	12
NC064		15
I3I2	$I3 \longrightarrow I2$	13
EN BF	ENABLE F RGTR	13
013	SET 013 FF	19

TERM NAME	COMMAND OR FUNCTION	DPD NO
NR100B	[PC=1]	19
SCLRF	CLR F RGTR	13
SFC00	$+1 \longrightarrow F$	
RTJ114		
NRX14A		19
ENABE	ENABLE E RGTR	13
ENABF	ENABLE F RGTR	13
NRJ100		19
SELBJ	SELECT BJ RGTR	2
NR114A		9
SELBI3	$B \longrightarrow I3$	12
NR100A	[PC≠1]	19
CLRF	CLR F RGTR	13
CLRF00	$0 \longrightarrow F_0$	13

TERM NAME	COMMAND OR FUNCTION	DPD NO
RTJ164		19
NRJXX4		19
I3I2	$I3 \longrightarrow I2$	13
ENABF	ENABLE F RGTR	13
RTJ164		19
NRTXX4		19
ENBP	ENABLE P RGTR	3
NR164A		19
SELKI3	$K \longrightarrow I3$	12
ADVPC2	ADVANCE PC CNTR	14

TERM NAME	COMMAND OR FUNCTION	DPD NO
RTJ214		19
NRX14A		19
ENABF	ENABLE F RGTR	13
RTJ214	[013 FF]	19
CLRRUN	CLR RUN FF	19
RTJ214		19
EXJREQ	$EXJREQ \longrightarrow XMIT$	19
	NOTE: "EXJREQ" BLOCKS RTJ264 SEQ TIMING	

CONTROL DATA CANADIAN DEVELOPMENT DIVISION	INSTRUCTION FLOW SEQUENCE: RETURN JUMP INSTRUCTION: 013	CODE IDENT 34570	D	DWG NO. 19981800 REV A
		FIGURE 5-2-6		PAGE NO 5-2-40.5

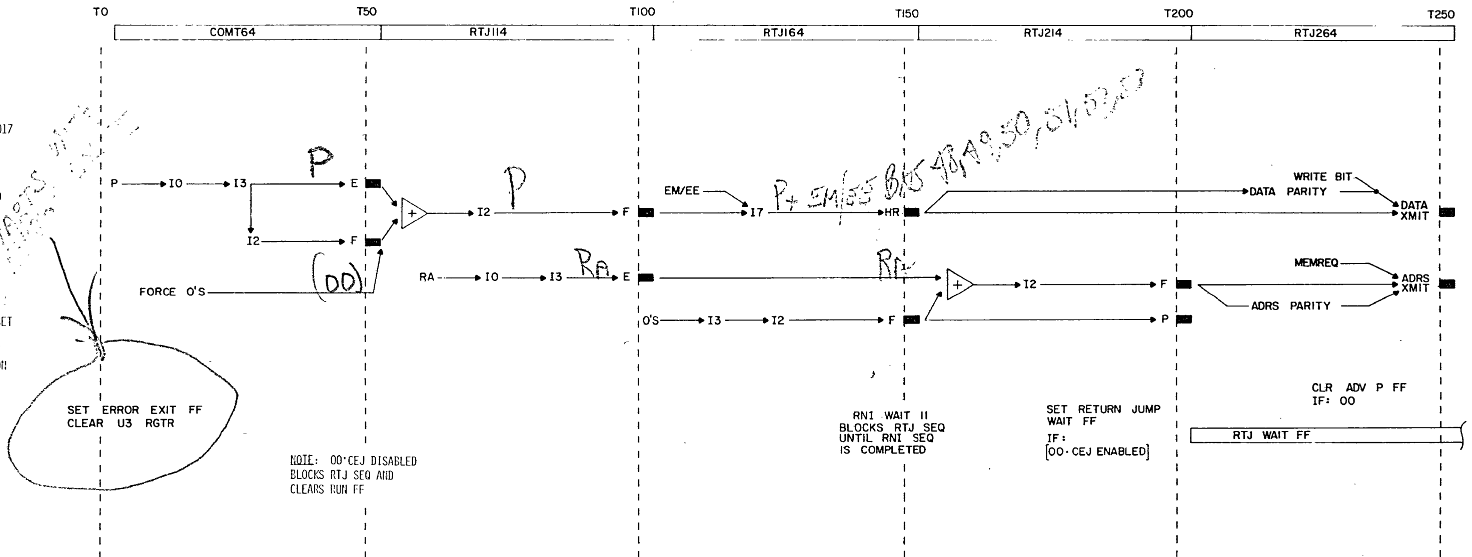
NOTES: AN "EE" SEQUENCE OCCURS UPON:

1) ILLEGAL INSTRUCTION
   - 013·PC=1
   - 013·CEJ DISABLED
   - 014 + 015 + 016 + 017
   - ECS ILLEGAL
   - CMU ILLEGAL
2) 30 BIT ILLEGAL
3) EXIT CONDITION SENSED
   - AØR
   - INDEFINITE
   - INFINITE
   - ADRS PARITY
   - DATA PARITY
   - DOUBLE ERROR
4) BREAKPOINT SENSED

ERROR EXIT FF (2.16) IS SET AT SEQUENCE EXIT. ERROR EXIT FF GENERATES "CLRU3" TO CLEAR U3 INSTRUCTION ON DECODE RGTR.

REF ADR IS CORR

STARTS



TERM NAME	COMMAND OR FUNCTION	DPD NO
NCØ64		15
SI00	P → I0	3
ENABE	ENABLE E RGTR	13
CØI0I3		15
SLI0I3	I0 → I3	12
NCØ64		15
I3I2	I3 → I2	13
ENABF	ENABLE F RGTR	13
GØRTJ	[00·CEJ DISABLED]	
CLRRUN	CLR RUN FF	19

TERM NAME	COMMAND OR FUNCTION	DPD NO
NR100A		19
CLRF	CLR F RGTR	13
CLRF00	0 → F	13
SI00·ST01 ·SI02	RA → I0	3
RTJ114		19
NRJX14	I0 → I3	19
SLI0I3		12
RTJ114		
NRX14A		19
ENABE	ENABLE E RGTR	13
ENABF	ENABLE F RGTR	13

TERM NAME	COMMAND OR FUNCTION	DPD NO
NR164B		19
SEEI7	EM/EE → $I7_{48-53}$	45
RTJ164		19
NRJXX4		19
I3I2	I3 → I2	13
ENABF	ENABLE F RGTR	13
RTJ150		
SFI7	F_{0-17} → $I7_{30-47}$	19
ENBHR	ENABLE HR RGTR	45

TERM NAME	COMMAND OR FUNCTION	DPD NO
RTJ214		
NRTXX4		19
ENBP	ENABLE P RGTR	3
RTJ214		19
NRX14A	ENABLE F RGTR	19
RTJ214		
RT214	[00·CEJ ENABLED]	19
RTJWAIT	SET RTJ WAIT FF	16

TERM NAME	COMMAND OR FUNCTION	DPD NO
NRJ264		19
CEMREQ		17
MEMREQ	MEMREQ → ADRS XMIT	17
ENBADT	CLOCK ADRS XMIT	17
NR264		19
ENBWRT		45
DWRITE	WRITE BIT → DATA XMIT	45
ENBDT	CLOCK DATA XMIT	45
RTJ264 SET 013_{FF}	{ [$00\cdot\overline{ERREXT}$ + $\overline{013\ FF}\cdot ERREXT$] SET 013 FF }	19
NR264		19
CLR ADV P FF	CLR ADV P FF	14

(Part 1 of 2)

CONTROL DATA CANADIAN DEVELOPMENT DIVISION

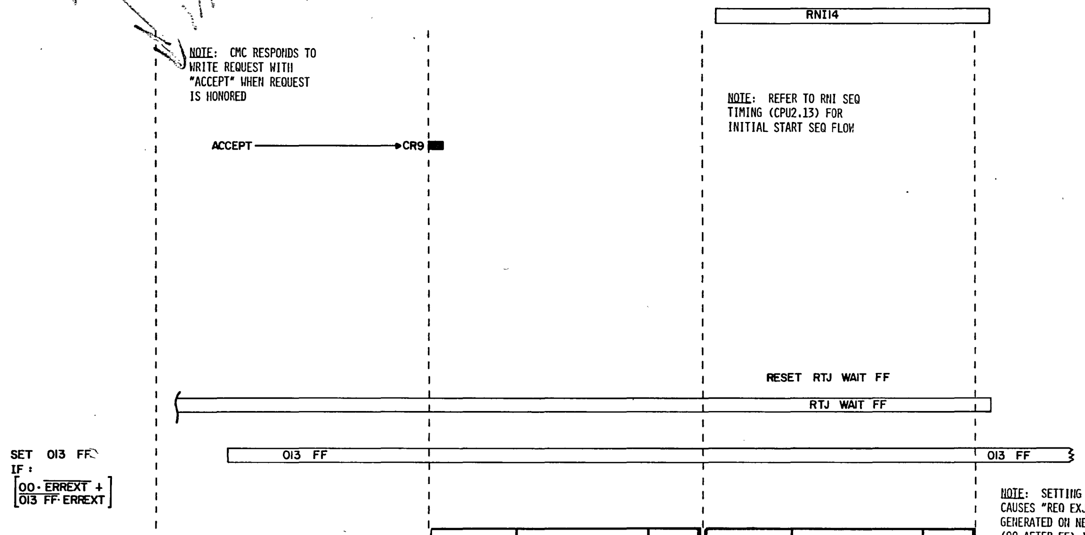
INSTRUCTION FLOW
SEQUENCE: RETURN JUMP
INSTRUCTION: ERROR EXIT, 00·$\overline{EE}$·CEJ ENABLED

CODE IDENT		DWG. NO.	REV
34570	D	19981800	A
FIGURE 5-2-7		PAGE NO 5-2-40·6	

WRITING ERROR CONDITION @ CR IN MONITOR

NOTE: CMC RESPONDS TO WRITE REQUEST WITH "ACCEPT" WHEN REQUEST IS HONORED

NOTE: REFER TO RNI SEQ TIMING (CPU2.13) FOR INITIAL START SEQ FLOW


RNI14
ACCEPT
CR9
RESET RTJ WAIT FF
RTJ WAIT FF
SET 013 FF
IF :
[00 · $\overline{\text{ERREXT}}$ + $\overline{\text{013 FF}}$ · ERREXT]
013 FF
013 FF

NOTE: SETTING OF 013 FF CAUSES "REQ EXJ" TO BE GENERATED ON NEXT RTN SEQ (00 AFTER EE) IF MONITOR FLAG CLEAR

TERM NAME	COMMAND OR FUNCTION	DPD NO
ACCEP2	[RTJ WAIT FF]	16
NRTJEX	ENABLE INITIAL START RNI SEQ	

TERM NAME	COMMAND OR FUNCTION	DPD NO
RNI14		
NRNE4A	CLR RTJ WAIT FF	16

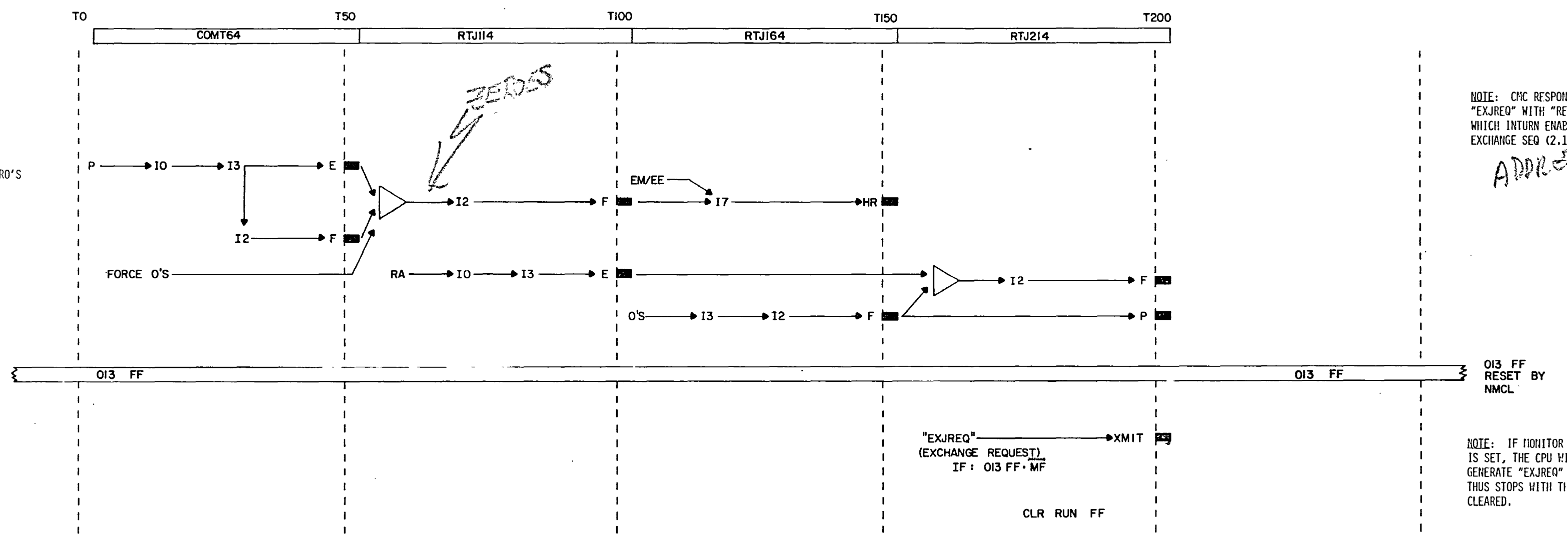
(Part 2 of 2)

CONTROL DATA CANADIAN DEVELOPMENT DIVISION	INSTRUCTION FLOW SEQUENCE: RETURN JUMP INSTRUCTION: ERROR EXIT, 00 · $\overline{\text{EE}}$ · CEJ ENABLED	CODE IDENT 34570	D	DWG. NO. 19981800 / REV A
		FIGURE 5-2-7		PAGE NO. 5-2-40·7

NOTE: P RGTR CLEARED TO ZERO'S DURING FIRST PASS RTJ SEQ

EXJ 013 FF SET DURING FIRST PASS RTJ SEQ BY

$\frac{00\cdot\overline{ERREXT}+}{\overline{013}\ FF\cdot ERREXT}$



013 FF RESET BY NMCL

NOTE: CMC RESPONDS TO "EXJREQ" WITH "REQEXJ" WHICH INTURN ENABLES EXCHANGE SEQ (2.19)

ADDRESS IS MA

NOTE: IF MONITOR FLAG (MF) IS SET, THE CPU WILL NOT GENERATE "EXJREQ" THE CPU THUS STOPS WITH THE RUN FF CLEARED.

TERM NAME	COMMAND OR FUNCTION	DPD NO
NCØ64		15
SI00	P → I0	3
ENABE	ENABLE E RGTR	13
CØ1013		15
SLI0I3	I0 → I3	12
NCØ64		15
I3I2	I3 → I2	13
ENABF	ENABLE F RGTR	13

TERM NAME	COMMAND OR FUNCTION	DPD NO
NR100A		19
CLRF	CLR F RGTR	13
CLRF00	0 → F	13
$\overline{SI00}\cdot\overline{SI01}$		
$\cdot\overline{SI02}$	RA → I0	3
RTJ114		19
NRJX14		19
SLI0I3	I0 → I3	12
RTJ114		19
NRX14A		19
ENABE	ENABLE E RGTR	13
ENABF	ENABLE F RGTR	13

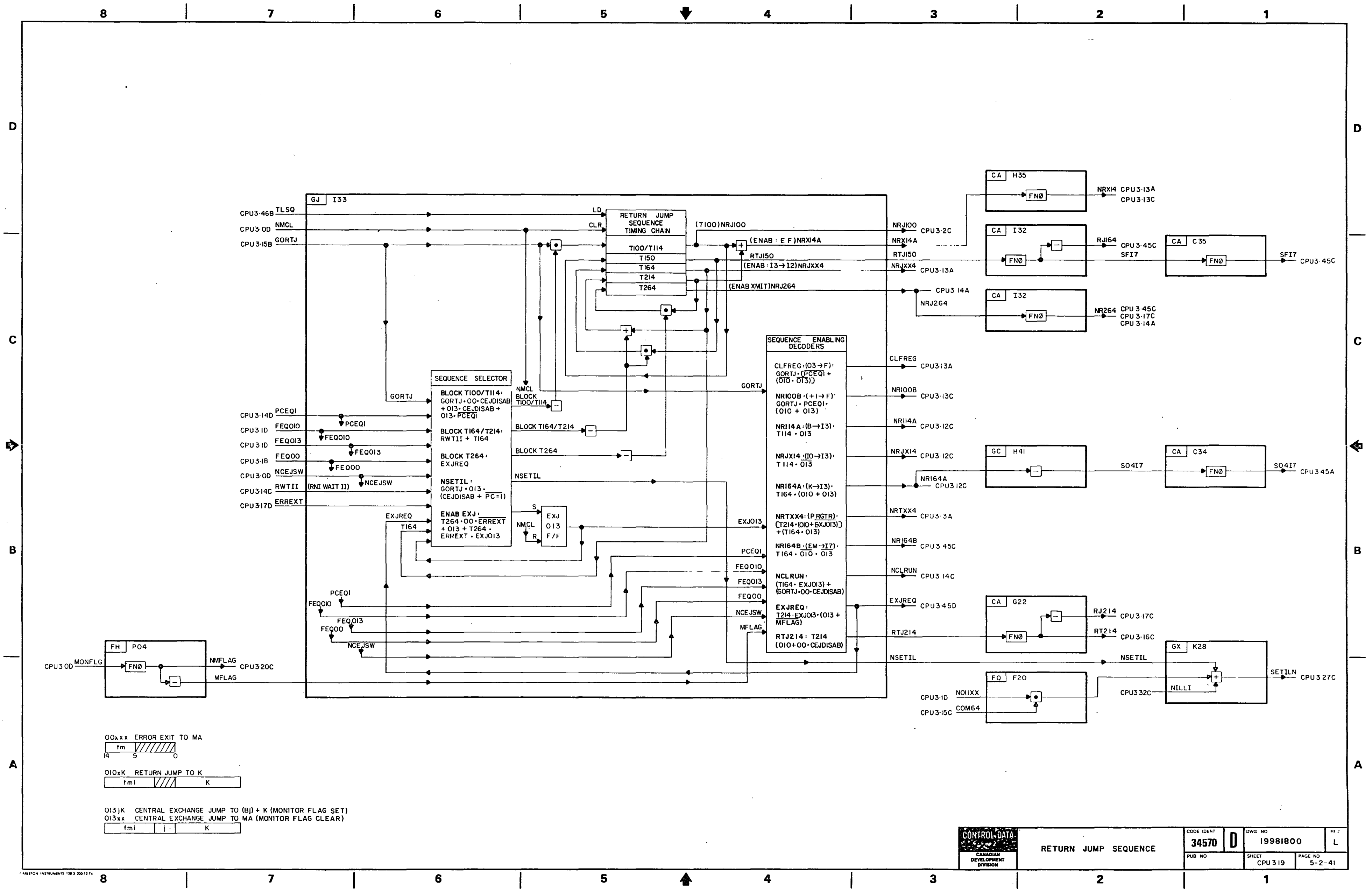
TERM NAME	COMMAND OR FUNCTION	DPD NO
NR164B		19
SEEI7	EM/EE → $I7_{48-53}$	45
RTJ164		19
NRJXX4		19
I3I2	I3 → I2	13
ENABF	ENABLE F RGTR	13
RTJ150		
SFI7	F_{0-17} → $I7_{30-47}$	19
ENBHR	ENABLE HR RGTR	

TERM NAME	COMMAND OR FUNCTION	DPD NO
RTJ214		
NRTXX4		19
ENBP	ENABLE P RGTR	3
RTJ214		19
NRX14A	ENABLE F RGTR	19
RTJ214	[013 FF]	19
CLRRUN	CLR RUN FF	19
RT214	[013 FF · $\overline{MFLAG}$]	19
EXJREQ	EXJREQ → XMIT	19
	NOTE: EXJREQ BLOCKS RTJ264 SEQ TIMING	

CONTROL DATA
CANADIAN DEVELOPMENT DIVISION

INSTRUCTION FLOW
SEQUENCE: RETURN JUMP
INSTRUCTION: 00. AFTER EE

CODE IDENT.		DWG. NO.	REV
34570	D	19981800	A
FIGURE 5 - 2 - 8		PAGE NO. 5-2-40-8	



CONTROL DATA CANADIAN DEVELOPMENT DIVISION

RETURN JUMP SEQUENCE

CODE IDENT		DWG NO	REV
34570	D	19981800	L
PUB NO		SHEET CPU3 19	PAGE NO 5-2-41

EXCHANGE SEQUENCE

An exchange jump can be issued from either the PPU or the CPU. An exchange jump interrupts the processor and causes it to exchange a 16-word package in central memory with the CPU registers. The package which the CPU receives contains all the initial operating parameters, and the package received by memory contains all the present operating parameters.

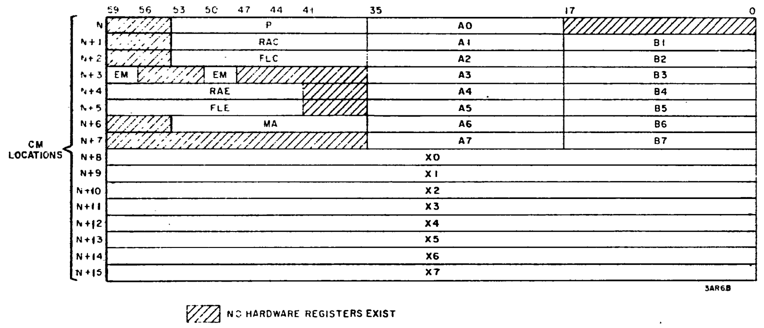
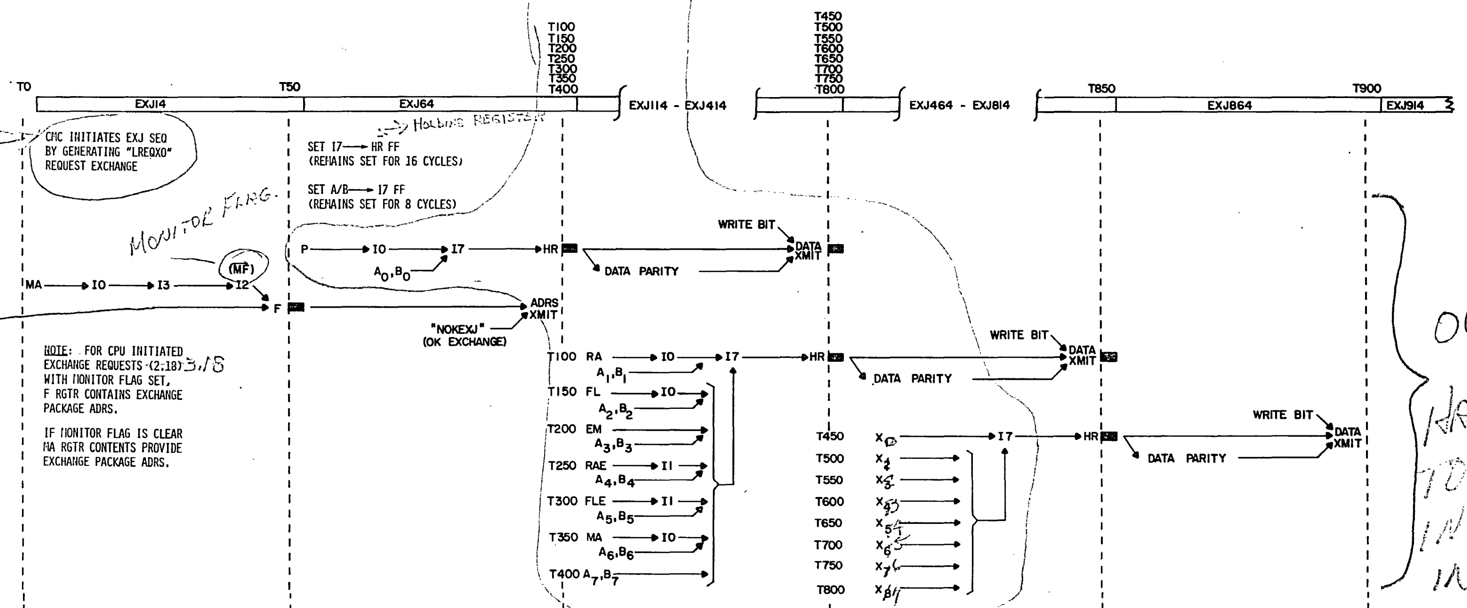


Figure 5-2-9. Exchange Package

The exchange sequence generates the necessary control signals to implement the exchange of data between the CPU and CMC. It also provides the internal controls to: enter the contents of the exchange package, interrupt the program currently being executed, and exchange the operating registers and control parameters with those of another program without information loss.

Whether the exchange request was first initiated by the CPU in response to a 013, 00/error exit instruction, or from the PPU, the exchange sequence is initiated by REQEXJ from CMC. An OK exchange signal (NOKEXJ) is returned to CMC when the parcel count = 0 at instruction exit or the CPU is stopped. These conditions ensure that all instructions of the last 60-bit word have been executed before initiating the exchange.

At the CPU, the exchange takes 1.4 usec to complete.



NOTE: FOR CPU INITIATED EXCHANGE REQUESTS ~~(2.18)~~ 3.18 WITH MONITOR FLAG SET, F RGTR CONTAINS EXCHANGE PACKAGE ADRS.

IF MONITOR FLAG IS CLEAR MA RGTR CONTENTS PROVIDE EXCHANGE PACKAGE ADRS.

FROM CMC TO CSU.

FROM PAGE 5-2-40.5

OLD PACKAGE GOING TO ADDRESS INCREMENT & HOLD CCT. ON PAGE 2.10 OF CMC

OLD PACKAGE HAS BEEN SENT TO CMC & [illegible] IN HOLD RGTR IN CMC

TERM NAME	COMMAND OR FUNCTION	DPD NO
LREQXO	REQUEST EXCHANGE	0
NEX14		20
SI01	MA —— I0	3
SLI0I3	I0 —— I3	12
NEX14A	[MONITOR FLAG]	20
I3I2	I3 → I2	12
ENABF	ENABLE F RGTR	13

TERM NAME	COMMAND OR FUNCTION	DPD NO
NEX50		20
A01·A02·A03	SELECT A0	2
B01·B02·B03	SELECT B0	2
1EX64	(T64·PASS 1)	20
SI00	P → I0	3
SI0I7	I0 → I7	45
ABI7	A/B → I7	20
EXI7HR	I7 → HR	45
ENBHR		
NOKEXJ	OK EXJ → XMIT	45
ENBADT	ENABLE ADRS XMIT	45

TERM NAME	COMMAND OR FUNCTION	DPD NO
NEX100-	SELECT A,-A7,	20
NEX400	B1-B7	2
1E114	(T114·PASS 1)	20
SI00·SI01		
·SI02	RA → I0	3
SI0I7	I0 → I7	45
1E164	(T164·PASS 1)	19
SI00·SI01	FL → I0	3
SI0I7	I0 → I7	45
1E214	(214·PASS 1)	20
SEEI7	EM I7	45
1E264	(T264·PASS 1)	20
SRAECS	RAE → I1	27
SRAS		4
SI1I7	I1 → I7	45
1E314	(T314·PASS 1)	20
SFLECS		27
SFLS	FLE I1	4
SI1I7	I1 I7	45
1E364	(T364·PASS 1)	20
SI01	MA → I0	3
SI0I7	I0 → I7	45
EXHRTR	HR DATA XMIT	45
DWRITE	WRITE BIT	
EXI7HR	I7 → HR	20
ENBHR		45

TERM NAME	COMMAND OR FUNCTION	DPD NO
NEX450-	SELECT X_0-X_7	20
NEX800		2
1EX464	(T464·PASS 1)	20
EXXI7		20
SLXI7	X → I7	1
CMXI7		45
EXI7HR		20
ENBHR	I7 HR	45
EXHRTR	HR → DATA XMIT	20
DWRITE	WRITE BIT	45

TERM NAME	COMMAND OR FUNCTION	DPD NO
EXHRTR	HR → DATA XMIT	20
DWRITE	WRITE BIT	45

(Part 1 of 2)

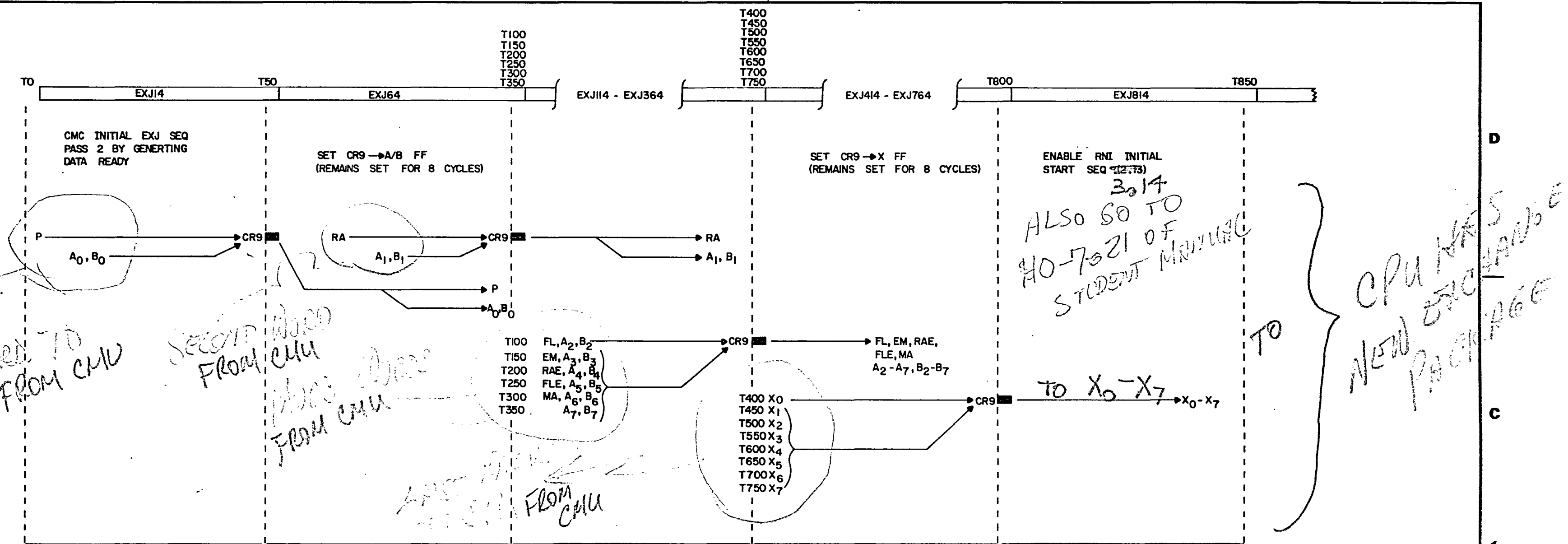
CONTROL DATA CANADIAN DEVELOPMENT DIVISION

INSTRUCTION FLOW SEQUENCE & EXCHANGE JUMP

CODE IDENT.		DWG. NO.	REV
34570	D	19981800	A
FIGURE 5-2-10		PAGE NO. 5-2-42-1	

CARLETON INSTRUMENTS 136-3 200-13-74

1/2 OF EXCHANGE JMP CYCLE


T0
EXJ14
T50
EXJ64
T100 T150 T200 T250 T300 T350
EXJ114 - EXJ364
T400 T450 T500 T550 T600 T650 T700 T750
EXJ414 - EXJ764
T800
EXJ814
T850
CMC INITIAL EXJ SEQ PASS 2 BY GENERTING DATA READY
SET CR9 → A/B FF (REMAINS SET FOR 8 CYCLES)
SET CR9 → X FF (REMAINS SET FOR 8 CYCLES)
ENABLE RNI INITIAL START SEQ ~~(2.73)~~ 3,14
P
A0, B0
CR9
RA
A1, B1
CR9
RA
A1, B1
P
A0, B0
T100 FL, A2, B2
T150 EM, A3, B3
T200 RAE, A4, B4
T250 FLE, A5, B5
T300 MA, A6, B6
T350 A7, B7
CR9
FL, EM, RAE, FLE, MA A2-A7, B2-B7
T400 X0
T450 X1
T500 X2
T550 X3
T600 X4
T650 X5
T700 X6
T750 X7
CR9
X0-X7
FIRST WORD TO CPU FROM CMU
SECOND WORD FROM CMU
FROM CMU
FROM CMU
ALSO GO TO 40-7021 OF STUDENT MANUAL
TO X0-X7
TO
CPU HAS NEW EXCHANGE PACKAGE

TERM NAME	COMMAND OR FUNCTION	DPD NO
DARDY	DATA READY	20
EXJ14	DATA HERE FROM CMU	

NEW EXCHANGE PACKAGE

TERM NAME	COMMAND OR FUNCTION	DPD NO
NEX50		20
A01·A02·A03	SELECT A0	2
B01·B02·B03	SELECT B0	2
2EX64	(T64·PASS2)	20
CR9P	$CR9 \xrightarrow[36-53]{} P$	20
ENBP		3
NCR9AB	CR9 → A/B FF	20
NCR9A		
NCR9B		
ENABA	ENABLE A RGTR	2
ENABB	ENABLE B RGTR	2

TERM NAME	COMMAND OR FUNCTION	DPD NO
NEX100-NEX350	SELECT A, -A6 B,-B6	20 2
2EX114	(T114·PASS 2)	20
ENBRA	$CR9 \xrightarrow[36-53]{} RA$	3
2EX164	(T164·PASS 2)	20
ENBFL	$CR9 \xrightarrow[36-53]{} FL$	3
ENABEM	(T214·PASS 2)	20
TLKEE	$CR9 \xrightarrow[48-50,\ 57-59]{} EM$	4
2EX264	(T264·PASS 2)	20
ENRAS	$CR9 \xrightarrow[42-59]{} RAE$	4
2EX314	(T314·PASS 2)	20
ENFLS	$CR9 \xrightarrow[42-59]{} FLE$	4
2EX364		20
ENBMA	$CR9 \xrightarrow[36-53]{} MA$	3

TERM NAME	COMMAND OR FUNCTION	DPD NO
NEX400-NEX750	SELECT A_7, B_7, X_0-X_6	20 2
NCR9X	(T464·PASS 2) CR9 → X FF	20
CR9X		2
NENBX	ENABLE X RGTR	2
ENABX		

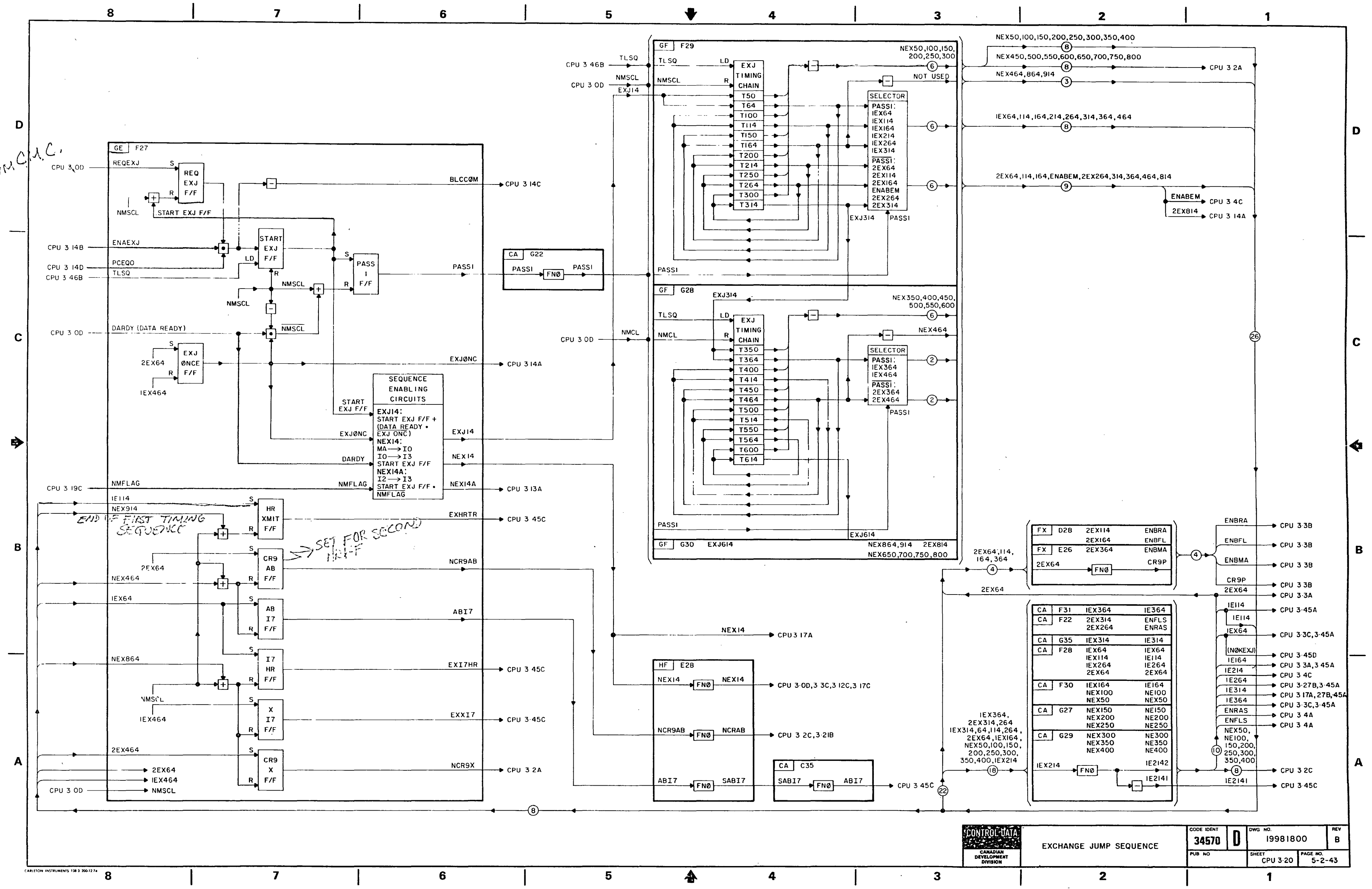
TERM NAME	COMMAND OR FUNCTION	DPD NO
NEX800	SELECT X_7	20 2
2EX814	(T814·PASS 2)	20
INSRT	RNI INITIAL START	14

(Part 2 of 2)

CONTROL DATA CANADIAN DEVELOPMENT DIVISION

INSTRUCTION FLOW SEQUENCE : EXCHANGE JUMP	CODE IDENT. 34570	D	DWG. NO. 19981800	REV A
	FIGURE 5-2-10		PAGE NO. 5-2-42-2	

2ND HALF OF EXCHANG JUMP



FROM C.M.C.
GE F27
REQ EXJ F/F
START EXJ F/F
PASS I F/F
EXJ ØNCE F/F
SEQUENCE ENABLING CIRCUITS
EXJ14: START EXJ F/F + (DATA READY • EXJ ONC)
NEX14: MA ⟶ I0, I0 ⟶ I3, START EXJ F/F
NEX14A: I2 ⟶ I3, START EXJ F/F • NMFLAG
HR XMIT F/F
CR9 AB F/F
AB I7 F/F
I7 HR F/F
X I7 F/F
CR9 X F/F
END OF FIRST TIMING SEQUENCE
SET FOR SECOND HR-F
BLCCØM
EXJØNC
EXJ14
NEX14
NEX14A
EXHRTR
NCR9AB
ABI7
EXI7HR
EXXI7
NCR9X
CA G22 PASSI FNØ PASSI
GF F29 EXJ TIMING CHAIN
T50 T64 T100 T114 T150 T164 T200 T214 T250 T264 T300 T314
SELECTOR PASSI: IEX64 IEX114 IEX164 IEX214 IEX264 IEX314 PASSI: 2EX64 2EX114 2EX164 ENABEM 2EX264 2EX314
GF G28 EXJ TIMING CHAIN
T350 T364 T400 T414 T450 T464 T500 T514 T550 T564 T600 T614
SELECTOR PASSI: IEX364 IEX464 PASSI: 2EX364 2EX464
GF G30 EXJ614
NEX50,100,150,200,250,300,350,400
NEX450,500,550,600,650,700,750,800
NEX464,864,914
IEX64,114,164,214,264,314,364,464
2EX64,114,164,ENABEM,2EX264,314,364,464,814
NOT USED
ENABEM
2EX814
HF E28 NEX14 FNØ NEX14 NCR9AB FNØ NCRAB ABI7 FNØ SABI7
CA C35 SABI7 FNØ ABI7
FX D28 2EX114 2EX164 ENBRA ENBFL
FX E26 2EX364 ENBMA 2EX64 FNØ CR9P
CA F31 IEX364 IE364
CA F22 2EX314 2EX264 ENFLS ENRAS
CA G35 IEX314 IE314
CA F28 IEX64 IEX114 IEX264 2EX64 IE64 IE114 IE264 2EX64
CA F30 IEX164 NEX100 NEX50 IE164 NE100 NE50
CA G27 NEX150 NEX200 NEX250 NE150 NE200 NE250
CA G29 NEX300 NEX350 NEX400 NE300 NE350 NE400
IEX214 FNØ IE2142 IE2141
CPU 3 0D
CPU 3 14B
CPU 3 14D
CPU 3 46B
CPU 3 19C
CPU 3 14C
CPU 3 14A
CPU 3 13A
CPU 3 45C
CPU 3 2A
CPU 3 4C
CPU 3 17A
CPU 3·0D,3 3C,3 12C,3 17C
CPU 3 2C,3·21B
CPU 3 3B
CPU 3 3A
CPU 3·45A
CPU 3·3C,3·45A
CPU 3·45D
CPU 3 3A,3 45A
CPU 3·27B,3·45A
CPU 3 17A,27B,45A
CPU 3 4A
CPU 3 2C

CONTROL DATA
CANADIAN DEVELOPMENT DIVISION

EXCHANGE JUMP SEQUENCE

CODE IDENT		DWG NO.	REV
34570	D	19981800	B
PUB NO		SHEET CPU 3·20	PAGE NO. 5-2-43

CARLETON INSTRUMENTS 138 3 200-12 74

DETAILED PAK DIAGRAM (CPU 3.21)

INCREMENT SEQUENCE

The increment sequence controls the ones complement addition and subtraction of 18-bit fixed point operands for increment instructions 50-77, and controls the formation of 60-bit ones complement sum and difference values for integer instructions 36 and 37.

INCREMENT INSTRUCTIONS

SET A:	50ijK	Set Ai to (Aj) + K
	51ijK	Set Ai to (Bj) + K
	52ijK	Set Ai to (Xj) + K
	53ijk	Set Ai to (Xj) + (Bk)
	54ijk	Set Ai to (Aj) + (Bk)
	55ijk	Set Ai to (Aj) - (Bk)
	56ijk	Set Ai to (Bj) + (Bk)
	57ijk	Set Ai to (Bj) - (Bk)

The 5x instructions perform a ones complement addition or subtraction of 18-bit operands. The 18-bit result is stored in address register Ai. Overflow is ignored, but an address range fault may result from overflow.

The first operand from the Aj, Bj or Xj register is selected at common time 64 and sent to the E register. For an operand selected from Xj, only the truncated lower 18 bits of the 60-bit word are sent to E.

The second operand selected at INC114 time from the Bk register or the K portion of the instruction itself (K = 18-bit signed constant) is sent to the F register. Selection of K causes the ADVPC2 signal to be generated (CPU 3-12) which, in turn, advances the parcel counter. The parcel counter will thus point to the next 15-bit instruction.

Addition or subtraction (by complement addition) of the two operands is performed through the F adder at INC164 time. The results are sent to the F register, and from F to the selected Ai register. If Ai $\neq 0$, a range test is performed on the quantity in the F register at INC214 time to determine if the program range limits have been exceeded.

A reference is made to central memory using the newly created absolute address plus the reference address from RA. The type of reference made is a function of the i designator value.

i = 0	No Memory Reference
i = 1-5	Read from Memory to Xi
i = 6, 7	Write into Memory from Xi

If the range test detected an address out of range condition, the following occurs independently of the exit mode selection.

If i = 1-5:	Xi is loaded with all zeros from CR9 and the contents of memory location Ai are unchanged.
If i = 6 or 7:	Xi retains its original contents and the contents of memory location Ai are unchanged.

SET B:	60ijK	Set Bi to (Aj) + K
	61ijK	Set Bi to (Bj) + K
	62ijK	Set Bi to (Xj) + K
	63ijk	Set Bi to (Xj) + (Bk)
	64ijk	Set Bi to (Aj) + (Bk)
	65ijk	Set Bi to (Aj) - (Bk)
	66ijk	Set Bi to (Bj) + (Bk)
	67ijk	Set Bi to (Bj) - (Bk)

The 6x instructions perform a ones complement addition or subtraction of 18-bit operands. The 18-bit result is stored in increment register Bi. An overflow condition is ignored.

Operands for the 6x instructions are obtained in the same manner as described for the 5x instructions. A memory reference is not performed.

SET X:	70ijK	Set Xi to (Aj) + K
	71ijK	Set Xi to (Bj) + K
	72ijK	Set Xi to (Xj) + K
	73ijk	Set Xi to (Xj) + (Bk)
	74ijk	Set Xi to (Aj) + (Bk)
	75ijk	Set Xi to (Aj) - (Bk)
	76ijk	Set Xi to (Bj) + (Bk)
	77ijk	Set Xi to (Bj) - (Bk)

The 7x instructions perform a ones complement addition or subtraction of 18-bit operands. The 18-bit result is stored in the lower 18 bits of operand register Xi. The sign of the result is extended to the upper 42 bits of operand register Xi. An overflow condition is ignored.

Operands for the 7x instructions are obtained in the same manner as described for the 5x instructions. A memory reference is not performed. Sign extension is performed by the complement I5 control.

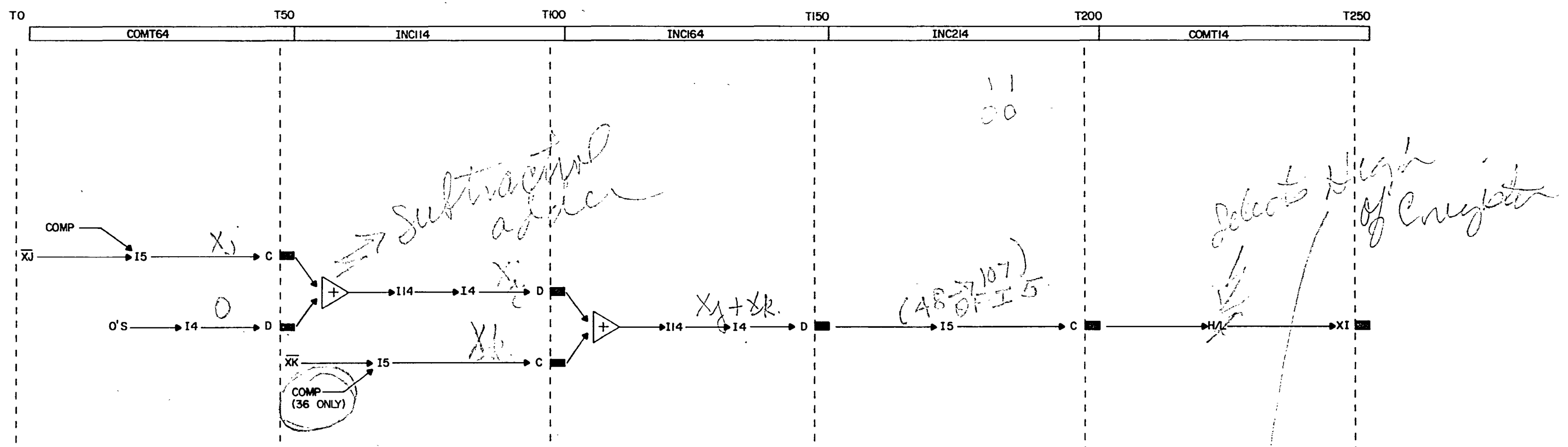
INTEGER SUM - DIFFERENCE

36ijk	Integer Sum of (Xj) and (Xk) to Xi
37ijk	Integer Difference of (Xj) and (Xk) to Xi

The 36 instruction forms a 60-bit ones complement sum of the quantities from operand registers Xj and Xk. Xj and Xk operands are considered as signed integers. Xj is sent to the D register, and Xk is sent to the C register. Integer addition is performed in the D adder, and the 60-bit result is stored in the Xi register. Overflow conditions are ignored.

The 37 instruction forms a 60-bit ones complement difference of the quantities from operand register Xj (minuend) and Xk (subtrahend). The 37 instruction allows the Xk complement to be sent to the C register; thus Xk is subtracted from Xj by complement addition through the D adder.

$\Longrightarrow$ COMPLEMENT & ADD



TERM NAME	COMMAND OR FUNCTION	DPD NO
CØN50		15
SELXJ	SELECT XJ RGTR	2
NCØ64		
I5185		7
I5285	$X_{0-59} \rightarrow I5_{48-107}$	7
I5164		7
I5264		7
I5236	$(I554 \cdot I552 \cdot \overline{I551})$	7
I5436		7
CØM50		15
ENABLC	ENABLE C RGTR	8
NCØ64		15
I40·I41	OS → I4	5
NCØ50		15
ENABD	ENABLE D RGTR	5

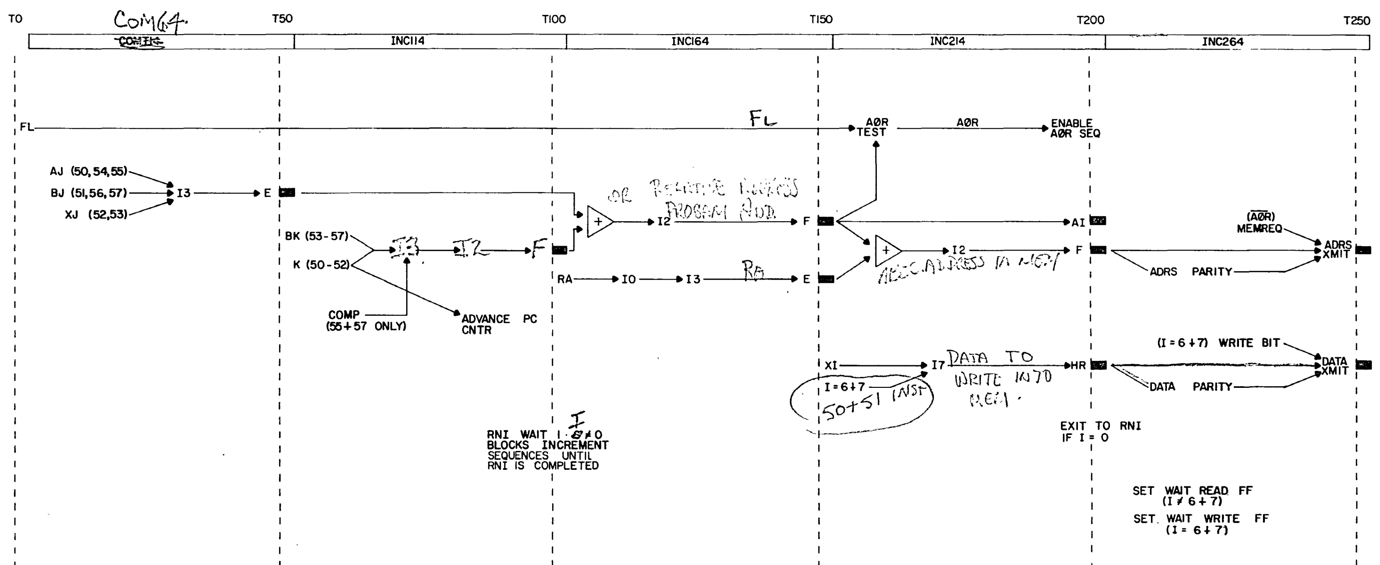
TERM NAME	COMMAND OR FUNCTION	DPD NO
$\overline{I14S}$	D ADD → I14	5
$\overline{I40} \cdot \overline{I41}$	I14 → I4	5
NINIX4		21
ENABD	ENABLE D RGTR	5
NI100		
SXK	SELECT XK RGTR	3
SELXK		3
NI114		21
I5185	$X_{0-59} \rightarrow I5_{48-107}$	7
I5285		7
I55236	$(I554 \cdot I552 \cdot \overline{I551})$	7
I55436		7
INC114	[FMEQ36] 3.13	21
ISCØ15	COMP I5	21
NINX14		
ENABLC	ENABLE C RGTR	8

TERM NAME	COMMAND OR FUNCTION	DPD NO
$\overline{I14S}$	D ADD → I14	5
$\overline{I40} \cdot \overline{I41}$	I14 → I4	5
NIN1X4		21
ENABD	ENABLE D RGTR	5

TERM NAME	COMMAND OR FUNCTION	DPD NO
INC214	[FM3637]	21
NI214A		21
I5085	$D_{48-107} \rightarrow I5_{48-107}$	7
I5064		7
I55136	$(\overline{I554} \cdot \overline{I552} \cdot I551)$	7
NINX14		
ENABLC	ENABLE C RGTR	8
NI214A		21
NINEXA	EXIT TO COMMON	21
CØMEXT	TIME, RNI	14
SEQEXT		14

TERM NAME	COMMAND OR FUNCTION	DPD NO
CØMX00		15
SELXI	SELECT XI RGTR	2
COMENX	ENABLE WRITE STROBE X RGTR	2
CØMX00		15
$\overline{HLSL}$	SELECT HIGHER	10

CONTROL DATA CANADIAN DEVELOPMENT DIVISION	INSTRUCTION FLOW SEQUENCE : INCR INSTRUCTION : 36, 37	CODE IDENT. 34570 D	DWG. NO. 19981800 REV A
		FIGURE 5-2-11	PAGE NO. 5-2-44-2



TERM NAME	COMMAND OR FUNCTION	DPD NO
CØM50		15
SELXJ	SELECT XJ RGTR	2
SELBJ	SELECT BJ RGTR	2
CØMT50		
SELAN	SELECT AJ RGTR	15
CØM64		
SELXI3	XI → I3	15
COMBI3	BI → I3	15
COMAI3	AI → I3	
NCØ50		
ENABE	ENABLE E RGTR	13

TERM NAME	COMMAND OR FUNCTION	DPD NO
NI100		21
SELBK	SELECT BK RGTR	21
IN114C	[53-57]	21
SELBI3	B ⟶ I3	12
IN114B	[50-52]	21
SELKI3		12
ADVPC2	{ K ⟶ I3	12
	ADVANCE PC COUNTER	
IN114A	[55+57]	21
SCØMI3	COMP I3	12
NI114		21
I3I2	I3 ⟶ I2	13
ENABF	ENABLE F RGTR	13

TERM NAME	COMMAND OR FUNCTION	DPD NO
$\overline{SI00}\cdot\overline{SI01}$		
$\cdot\overline{SI02}$	RA ⟶ I0	3
NI164		
SLI0I3	I0 ⟶ I3	12
ENABE	ENABLE E RGTR	13
NINX64		21
ENABF	ENABLE F RGTR	13

TERM NAME	COMMAND OR FUNCTION	DPD NO
NINX64		21
ENABF	ENABLE F RGTR	13
NI214C	[I + 0]	21
ENABØR	ENABLE TEST AØR	17
NI200		21
SELAI	SELECT AI RGTR	2
NCR9A	F ⟶ A	2
INC214		21
ENADA	ENABLE WRITE STROBE A RGTR	21
NI200		21
SELXI	SELECT XI RGTR	2
I214D	[I=6+7]	21
$\overline{CNI7}\cdot$CNX7	X ⟶ I7	45
I214D	[I=6+7]	21
ENBHR	ENABLE HR RGTR	45
INC214	[I=0]	21
NINEXB	EXIT TO RNI	21

TERM NAME	COMMAND OR FUNCTION	DPD NO
NIN264		21
CEMREQ	[AØR]	17
MEMREQ	MEMREQ ⟶ ADRS XMIT	17
ENBADT	CLOCK ADRS XMIT	17
NI264A	L=6+7	21
ENWDRT		45
BWRITE	WRITE BIT ⟶ DATA XMIT	45
		45
ENBDT	CLOCK DATA XMIT	45
IN264A	I=6+7	
SET WAIT READ FF	SET WAIT READ FF	16
IN264B	$I=\overline{6}+\overline{7}$	
SET WAIT WRITE FF	SET WAIT WRITE FF	16

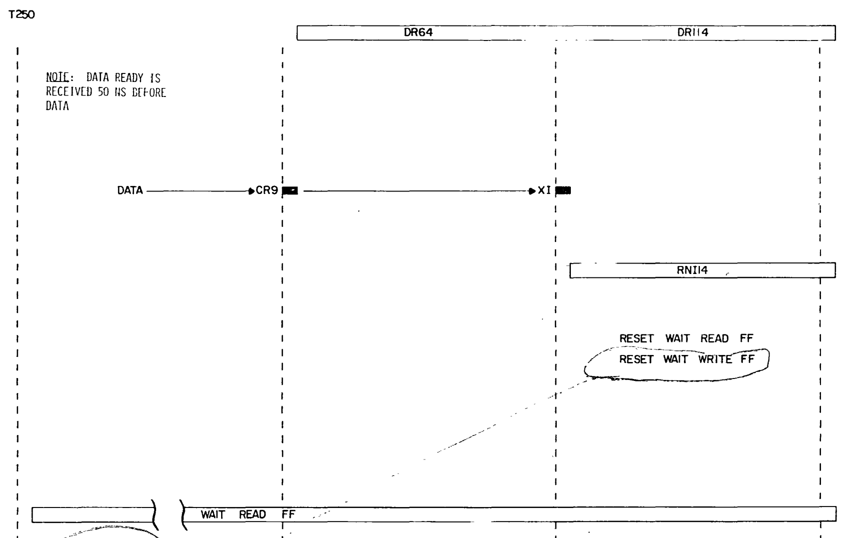
(Part 1 of 2)

CONTROL DATA CANADIAN DEVELOPMENT DIVISION

INSTRUCTION FLOW
SEQUENCE : INCR
INSTRUCTION : 50-57

CODE IDENT		DWG NO	REV
34570	D	19981800	A
FIGURE 5-2-12		PAGE NO 5-2-44-3	

CARLETON INSTRUMENTS 138-3 200-12-74



<u>NOTE</u>: DATA READY IS RECEIVED 50 NS BEFORE DATA

<u>NOTE</u>: FOR WRITE REQUEST CPU WAITS UNTIL CMC SENDS ACCEPT RESPONSE AT WHICH TIME "WREXT" IS GENERATED FROM GM MODULE (2.15).

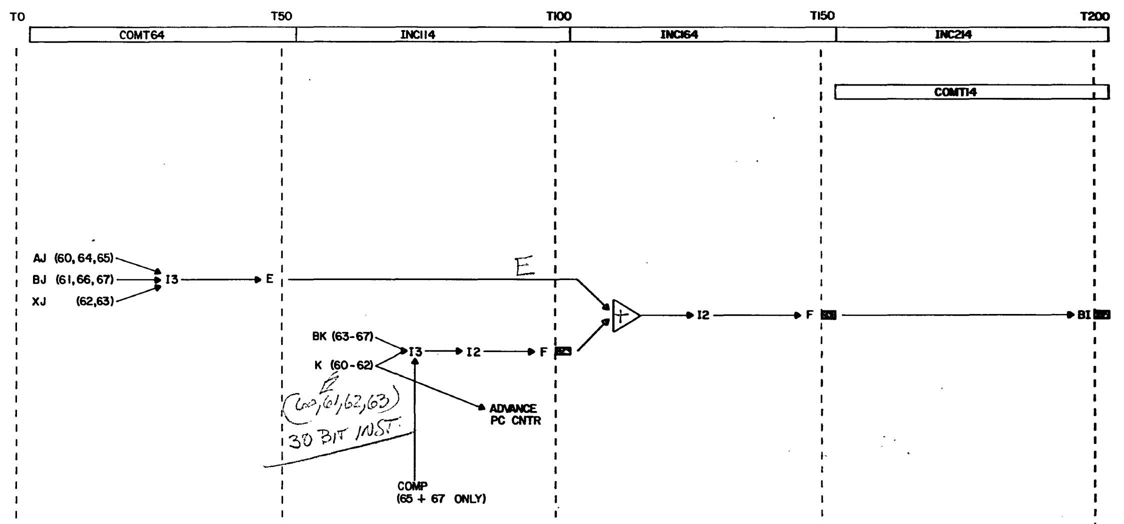
"WREXT" (WRITE EXIT) ENABLES SEQUENCE EXIT.

TERM NAME	COMMAND OR FUNCTION	DPD NO
NDR50	[DATA READY]	16
SELXI	SELECT XI RGTR	2
DRG4	[WAIT READ FF]	16
NWRDA		16
CR9X	CR9 $\longrightarrow$ X	2
NENBX	ENABLE WRITE	2
ENABX	STROBE X RGTR	2
NWRD4		16
SEQEXT	ENABLE EXIT TO RNI	14

TERM NAME	COMMAND OR FUNCTION	DPD NO
RNI14		
NRNE4A	CLR WAIT READ WAIT WRITE FF	14

(Part 2 of 2)

CONTROL DATA CANADIAN DEVELOPMENT DIVISION	INSTRUCTION FLOW SEQUENCE : INCR INSTRUCTION : 50-57	CODE IDENT 34570	D	DWG NO 19981800 REV A
		FIGURE 5-2-12		PAGE NO. 5-2-44.4



TERM NAME	COMMAND OR FUNCTION	DPD NO
CØM50		15
SELXJ	SELECT XJ RGTR	2
SELBJ	SELECT BJ RGTR	2
CØMT50		
SELAJ	SELECT AJ RGTR	15
CØM64		
SELXI3	XI → I3	15
CØMBI3	BI → I3	15
CØMAI3	AI → I3	15
NCØ50		
ENABE	ENABLE E RGTR	13

TERM NAME	COMMAND OR FUNCTION	DPD NO
NI100		21
SELBK	SELECT BK RGTR	21
IN114C	[63-67]	21
SELBI3	B → I3	12
IN114B	[60-62]	21
SELKI3	K → I3	12
ADVPC2	ADVANCE PC CNTR	12
IN114A	~~[55+57]~~ [65+67]	21
SCOMI3	COMP I3	12
NI114		21
I3I2	I3 → I2	13
ENABF	ENABLE F RGTR	13

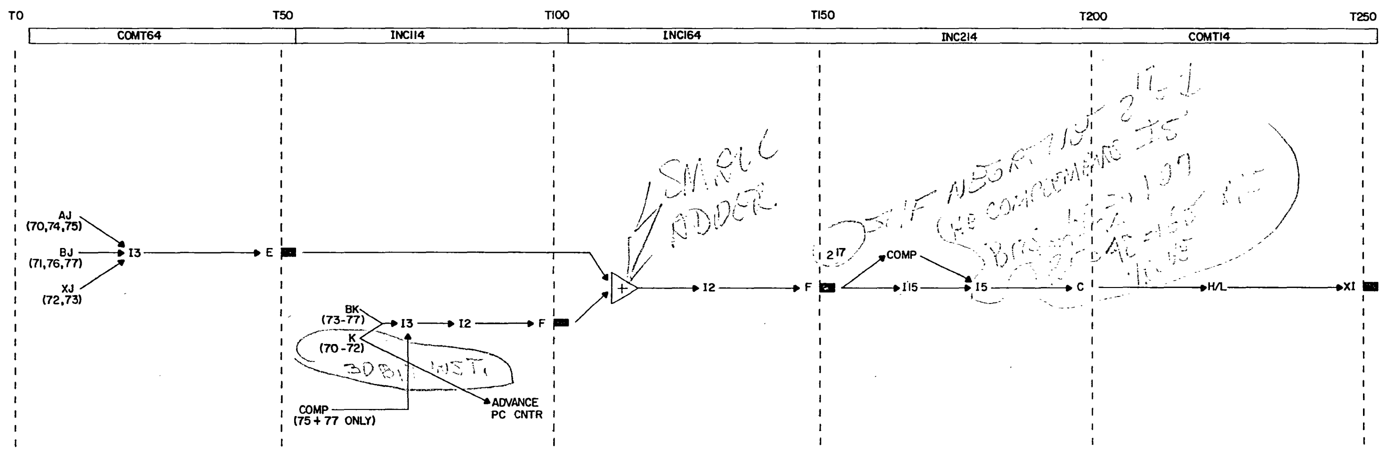
TERM NAME	COMMAND OR FUNCTION	DPD NO
NINX64	ENABLE F RGTR	21
NI164		21
NINEXA	EXIT TO COMMON	21
CØMEXT	TIME, RNI	14
SEQEXT		14

TERM NAME	COMMAND OR FUNCTION	DPD NO
NI200		21
SELBI	SELECT BI RGTR	2
CØMX00		15
COMENB	ENABLE WRITE STROBE B RGTR	2

INSTRUCTION FLOW
SEQUENCE : INCR
INSTRUCTION : 60, 67

CODE IDENT.		DWG. NO.	REV
34570	D	19981800	A
FIGURE 5-2-13		PAGE NO. 5-2-44-5	

60 THRU 67



TERM NAME	COMMAND OR FUNCTION	DPD NO
CØM50		15
SELXJ	SELECT XJ RGTR	2
SELBJ	SELECT BJ GRTR	2
CØMT50		
SELAJ	SELECT AJ RGTR	15
CØM64		
SELXI3	XI → I3	15
COMBI3	BI → I3	15
COMAI3	AI → I3	15
NCØ50		
ENABE	ENABLE E RGTR	13

TERM NAME	COMMAND OR FUNCTION	DPD NO
NI100		21
SELBK	SELECT BK RGTR	21
IN114C	[73-77]	21
SELBI3	B → I3	12
IN114B	[70-72]	21
SELKI3	K → I3	12
ADVPC2	ADVANCE PC CNTR	12
I114A	[75+77]	21
SCØMI3	COMP I3	12
NI114		21
I3I2	I3 → I2	13
ENABF	ENABLE F RGTR	13

TERM NAME	COMMAND OR FUNCTION	DPD NO
NINX64	ENABLE F RGTR	21

TERM NAME	COMMAND OR FUNCTION	DPD NO
NI214B	[FUN7]	21
NINEXA	EXIT TO COMMON	21
CØMEXT		14
SEQEXT	TIME, RNI	14
$\overline{\text{F46X}}$	$F_{0-17} \rightarrow$ I15	28
NI214B	[FUN7]	21
I5285	I15 → $I5_{48-65}$	7
	$(I5S4 \cdot \overline{I5S2} \cdot \overline{I5S1})$	
NI214B		21
NINX14	ENABLE C RGTR	8
IN214A	[F17]	21
ISCØI5	COMP I5	21

TERM NAME	COMMAND OR FUNCTION	DPD NO
CØMX00		15
SELXI	SELECT XI RGTR	2
CØMX00	ENABLE WRITE STROBE	15
CØMENX	X RGTR	15
$\overline{\text{HLSL}}$	SELECT HIGHER	10

CONTROL DATA
CANADIAN DEVELOPMENT DIVISION

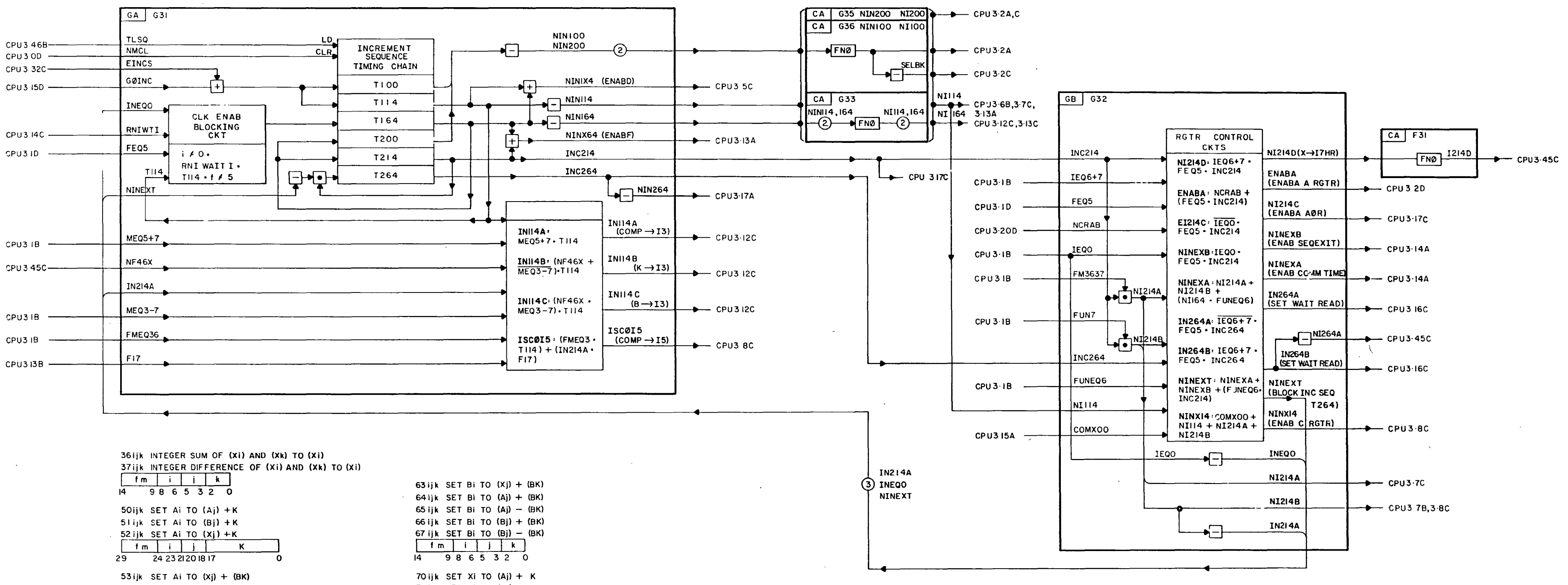
INSTRUCTION FLOW
SEQUENCE : INCR
INSTRUCTION : 70 - 77

CODE IDENT 34570 | D | DWG NO 19981800 | REV A

FIGURE 5 - 2 - 14 | PAGE NO 5-2-44-6

CARLETON INSTRUMENTS

70 THRU 77



36ijk INTEGER SUM OF (Xi) AND (Xk) TO (Xi)
37ijk INTEGER DIFFERENCE OF (Xi) AND (Xk) TO (Xi)

fm	i	j	k
14 9	8 6	5 3	2 0

50ijk SET Ai TO (Aj) +K
51ijk SET Ai TO (Bj) +K
52ijk SET Ai TO (Xj) +K

fm	i	j	K
29 24	23 21	20 18	17 0

53ijk SET Ai TO (Xj) + (BK)
54ijk SET Ai TO (Aj) + (BK)
55ijk SET Ai TO (Aj) − (BK)
56ijk SET Ai TO (Bj) + (BK)
57ijk SET Ai TO (Bj) − (BK)

fm	i	j	k
14 9	8 6	5 3	2 0

60ijk SET Bi TO (Aj) + K
61ijk SET Bi TO (Bj) + K
62ijk SET Bi TO (Xj) + K

fm	i	j	K
29 24	23 21	20 18	17 0

63ijk SET Bi TO (Xj) + (BK)
64ijk SET Bi TO (Aj) + (BK)
65ijk SET Bi TO (Aj) − (BK)
66ijk SET Bi TO (Bj) + (BK)
67ijk SET Bi TO (Bj) − (BK)

fm	i	j	k
14 9	8 6	5 3	2 0

70ijk SET Xi TO (Aj) + K
71ijk SET Xi TO (Bj) + K
72ijk SET Xi TO (Xj) + K

fm	i	j	K
29 24	23 21	20 18	17 0

73ijk SET Xi TO (Xj) + (BK)
74ijk SET Xi TO (Aj) + (BK)
75ijk SET Xi TO (Aj) − (BK)
76ijk SET Xi TO (Bj) + (BK)
77ijk SET Xi TO (Bj) − (BK)

fm	i	j	k
14 9	8 6	5 3	2 0

CONTROL DATA CANADIAN DEVELOPMENT DIVISION	INCREMENT SEQUENCE	CODE IDENT 34570	D	DWG NO 19981800	REV B
		PUB NO		SHEET CPU3-21	PAGE NO. 5-2-45

CARLETON INSTRUMENTS 138 3 200-12 74

DETAILED PAK DIAGRAM (CPU 3.22)

SHIFT SEQUENCE

The shift sequence controls the operations necessary to perform the following instructions:

20ijk	Left Shift (Xi) by jk
21ijk	Right Shift (Xi) by jk
22ijk	Left Shift (Xk) Nominally (Bj) Places to Xi
23ijk	Right Shift (Xk) Nominally (Bj) Places to Xi
24ijk	Normalize (Xk) to Xi and Bj
25ijk	Round Normalize (Xk) to Xi and Bj
26ijk	Unpack (Xk) to Xi and Bj
27ijk	Pack (Xk) and (Bj) to Xi
43ijk	Form Mask of jk Bits to Xi

SHIFT 20, 21

The 20 instruction reads the selected Xi operand and shifts the 60-bit word left circularly by jk bit positions. The bits which are shifted off the upper end are inserted in the lowest order bit positions.

The 21 instruction reads the selected Xi operand and shifts the 60-bit word right with sign extension by jk bit positions.

NOMINAL SHIFT 22, 23

The 22 instruction reads the selected Xk operand and shifts the 60-bit word either left or right as specified by (Bj). If (Bj) is positive, the data is shifted left circularly by the number of bit positions designated by (Bj). If (Bj) is negative, the data is shifted right with sign extension by the ones complement of the number of bit positions designated by (Bj).

The 23 instruction operates in a manner similar to a 22 instruction except that if (Bj) is positive right shifts are performed, and if (Bj) is negative left shifts are performed.

When shifting right, if the shift count in F is > 177_8, gating of the shift network to I5 during SH264 is blocked. A result of zeros is sent to the Xi register.

NORMALIZE 24, 25

The normalize instruction reads the selected Xk operand and performs a normalize operation on this word, delivering the normalized result back to the Xi register and the normalize count to the Bj register.

Normalization involves left shifting the coefficient until bit 47 is different from the coefficient sign bit. The exponent is decreased by the number of bit positions shifted. The normalize count used to shift the coefficient is developed by the normalize network. The normalize count is sent to the SK register during SH164 to enable the desired shift; it is also sent to the F register for subsequent writing into Bj during common time.

At the beginning of the normalize instruction, the Xj exponent is checked for indefinite or infinite operands. An indefinite or infinite operand causes the Xk operand to be returned to Xj unchanged, and gates zeros to Bj.

The normalize instruction also checks for exponent underflow after the normalize count is subtracted. If underflow is detected, the C register is cleared to zeros before initiating common time. The resulting operand sent to the Xi register will contain a zero exponent and coefficient.

The 25 instruction operates in a manner similar to the 24 instruction, except that bit 107 is set in the C register before sending C to the shift network. This round bit has the effect of increasing the magnitude of the coefficient by one half the value of the least significant bit, after the shift is performed.

In addition to checking for underflow, the 24 instruction checks for a coefficient equal to zero. The end case result, when coefficient equal to zero is detected, is the same as underflow. (See table 5-2-16.)

if Bj = 77775 complement the shift count equals 000002 the X reg. is then shifted right 2 places.

TABLE 5-2-16. OVERFLOW AND UNDERFLOW CONDITIONS

OVERFLOW		
INSTRUCTIONS	OVERFLOW CONDITION	RESULT
Normalize (24, 25)	None	---
UNDERFLOW		
INSTRUCTIONS	UNDERFLOW CONDITION	RESULT
Normalize (24 only)	Initial coefficient = ± 0	X_i = 0000 0....0_8, (Bj) = 60_8
Normalize (24, 25)	Final Exponent ≤ -2000_8	X_i = 0000 0....0_8, (Bj) are correct (See Note below.)
Note: Underflow of Exponent During Normalization: The final (Bj) are the same as if underflow had not occurred. In particular, if the initial coefficient is zero, (Bj) are equal to 60_8.		

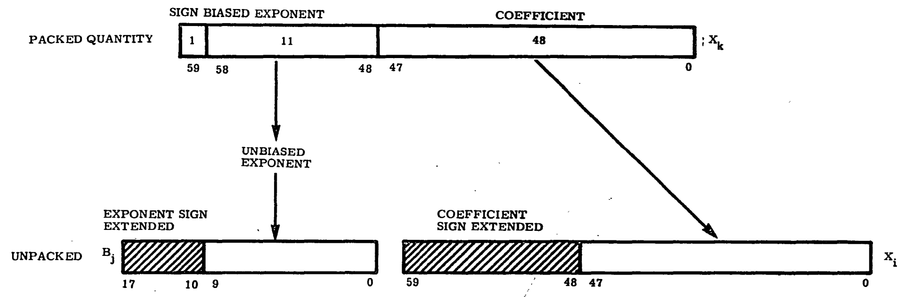
Error Exit Conditions

If Xk contains an infinite quantity (3777 x....x_8 or 4000 x....x_8) or an indefinite quantity (1777 x....x_8 or 6000 x....x_8), an optional exit mode selection is provided. The CPU response is dependent on whether the appropriate exit mode selection was made and the monitor flag /MEJ/CEJ condition.

An exit condition sensed (ECONDS) sets the ERROR EXIT FF (3.17) at the same time as the next RNI sequence is initiated. Error exit clears the U3 instruction register, thus forcing a return jump error exit sequence.

UNPACK, PACK 26, 27

The 26 instruction reads the selected Xk operand, unpacks this word from the floating point format, and delivers the coefficient to the Xi register and the exponent to the Bj register.



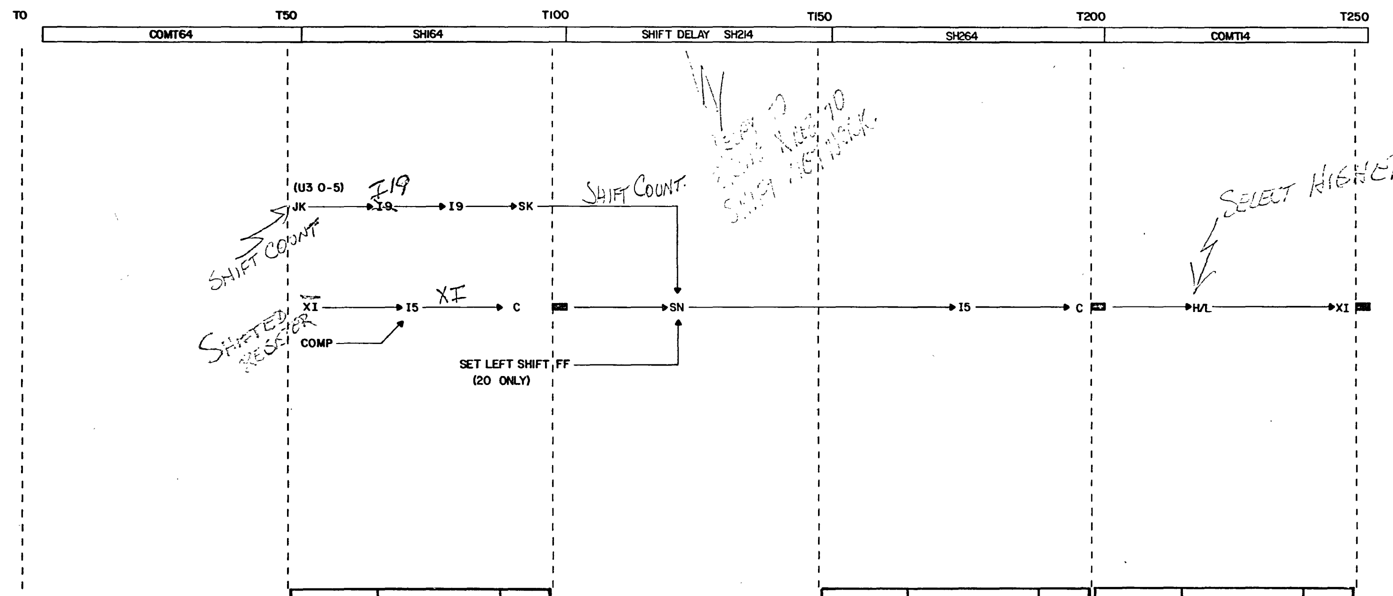
The 60-bit word delivered to the Xi register during common time (COMT00) consists of the lowest 48 bits unaltered from Xk, plus 12 bits equal to the sign bit.

The 18-bit quantity delivered to the Bj register during common time (COMT00) consists of the Xk exponent unbiased and sign extended. Unbiasing the exponent and sign extension is performed through I3 during SH114.

The 27 instruction performs the reciprocal process of the 26 instruction. The unpacked quantities in Xk and Bj are packed in floating point format and delivered to the Xi register.

MASK 43

The 43 instruction generates a masking word using the 6-bit jk quantity to designate the width of the masking field. The quantity is sent to the SK register. The C register is cleared to zero and sent to the shift network. During the shift period, C is right shifted by the jk quantity in SK. One-bits are forced to the shift network sign extension scheme, thus replacing each shifted zero bit with a one-bit. The completed masking word sent to the Xi register consists of one-bits in the highest order jk bit positions, and zero bits in the remainder of the word.



TERM NAME	COMMAND OR FUNCTION	DPD NO
SH150		22
SHSLHI	SELECT XI RGTR	22
F46X	$U3_{0-5} \rightarrow I19$	10
SH164		22
SLI90·SLI91	$I19 \rightarrow I9$	9
NSH164		22
SKS1·SKS2	$I9 \rightarrow SK$	9
SH164		
SI5185		22
SI5285	$X_{0-59} \rightarrow I5_{48-107}$	22
SI5164		22
SI5157	$(I5S4 \cdot I5S2 \cdot \overline{I5S1})$	22
SI5267		22
SH164	[FFUNN4]	
BLI5C	COMP I5	21
SH164		
SHENBC	ENABLE C RGTR	22
SH164	[20]	
RS	SET LEFT SHIFT FF	22

TERM NAME	COMMAND OR FUNCTION	DPD NO
SH264		
SI5085		22
SI5285	$SN_{48-107} \rightarrow I5_{48-107}$	22
SI5064		22
SI5057		22
SI5267	$(I5S4 \cdot \overline{I5S2} \cdot I5S1)$	22
SH264		
SHENBC	ENABLE C RGTR	22
SH264		
SHNEXT	ENABLE COMMON TIME,RNI	22

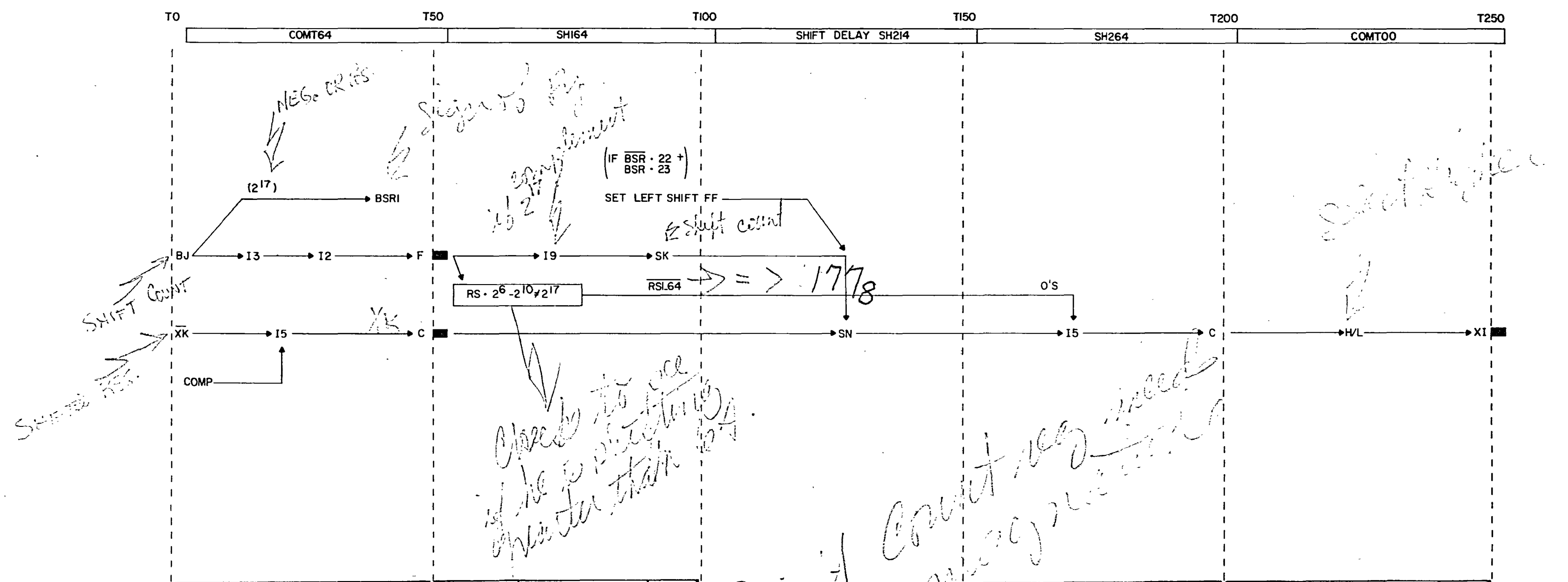
TERM NAME	COMMAND OR FUNCTION	DPD NO
CØMX00		15
SELXI	SELECT XI RGTR	2
CONRNX	ENABLE WRITE STROBE X RGTR	2
CØMX00		15
HLSL	SELECT HIGHER	10

CONTROL DATA CANADIAN DEVELOPMENT DIVISION

INSTRUCTION FLOW
SEQUENCE: SHIFT
INSTRUCTIONS: 20,21

CODE IDENT.		DWG. NO.	REV
34570	D	19981800	A
FIGURE 5-2-15		PAGE NO. 5-2-46-3	

CARLETON INSTRUMENTS 138-3 300-12-74



TERM NAME	COMMAND OR FUNCTION	DPD NO
CØM50	[FFE02]	2
SELXK	SELECT XK RGTR	2
CØM50		
SELBJ	SELECT BJ RGTR	2
COMT64		
COMBI3	$B \longrightarrow I3$	15
NCØ64		
I3I2	$I3 \longrightarrow I2$	13
ENABF	ENABLE F RGTR	13
CØMT64	[IF B17=1]	
BSRI	SET BSRI	24
NCØ64		
I5185		7
I5285	$X_{0-59} \longrightarrow I5_{48-107}$	7
I5164		7
I5264		7
I5236	$(I5S4 \cdot I5S2 \cdot \overline{I5S1})$	7
I5436		7
CØM64		
I5C85	COMP $I5_{48-107}$	8
I5C67		8
NCØ50		
ENABLC	ENABLE C RGTR	8

TERM NAME	COMMAND OR FUNCTION	DPD NO
SH1€4		
SHI9-1·SHI9-0	$F_{0-6} \longrightarrow I9$	22
NSH164		
SKS1·SKS2	$I9 \longrightarrow SK$	9
RS	[F 6-10=F17]	22
RSLT64	SHIFT RIGHT LESS THAN 64_{10}	22

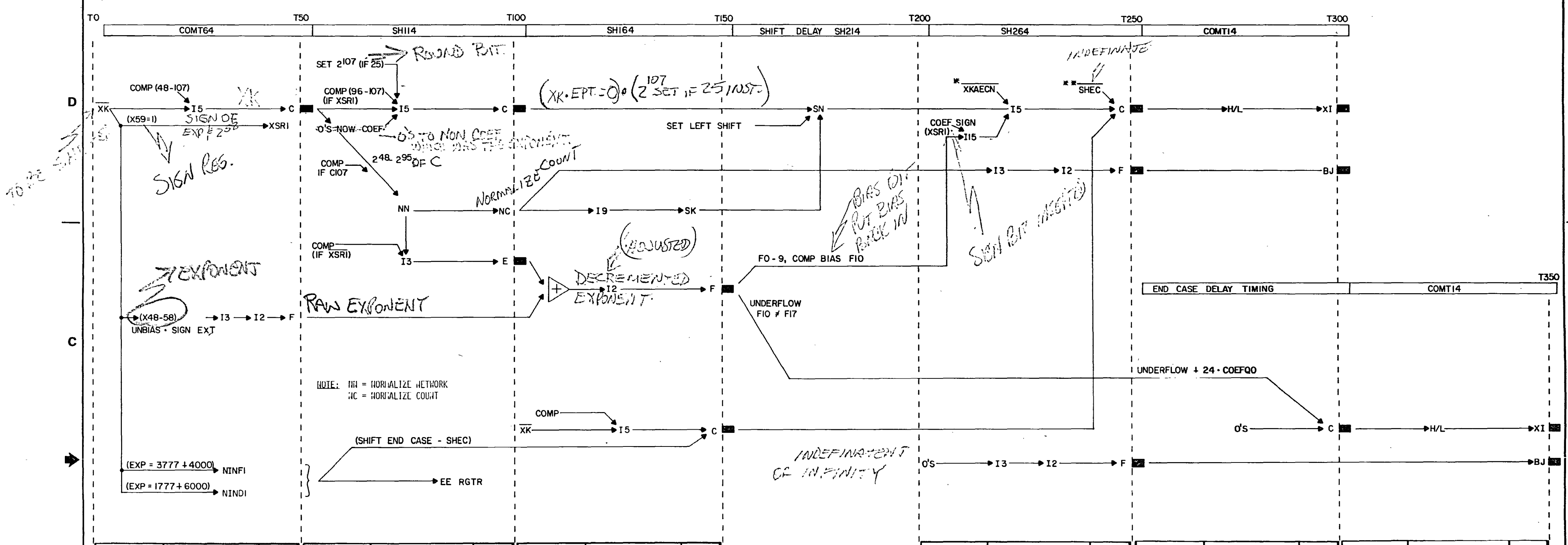
TERM NAME	COMMAND OR FUNCTION	DPD NO
SH264	[RSLT64]	
SI5085		22
SI5285	$SN_{\overline{48-107}} \longrightarrow I5_{48-107}$	22
SI5064		22
SI5057	$(I5S4 \cdot \overline{I5S2} \cdot I5S1)$	22
SI5267		22
SH264		
SHENBC	ENABLE C RGTR	22
SH264		
SHNEXT	ENABLE COMMON TIME, RNI	22

TERM NAME	COMMAND OR FUNCTION	DPD NO
COMX00		15
SELXI	SELECT XI RGTR	2
COMENX	ENABLE WRITE STROPE X RGTR	2
COMX00		15
$\overline{HLSL}$	SELECT HIGHER	10

CONTROL DATA
CANADIAN DEVELOPMENT DIVISION

INSTRUCTION FLOW
SEQUENCE: SHIFT
INSTRUCTION: 22, 23

CODE IDENT		DWG NO	REV
34570	D	19981800	A
FIGURE 5-2-16		PAGE NO 5-2-46·4	



TERM NAME	COMMAND OR FUNCTION	DPD NO
CØM50		15
SELXK	SELECT XK RGTR	2
CØM64		15
I5C85	COMPI5 48-95	8
I5C67	COMP I5 96-107	8
NCØ64		15
I5185		7
I5285	$X_{0-59} \rightarrow I5_{48-107}$	7
I5164		7
I5264	$(I5S4 \cdot I5S2 \cdot \overline{I5S1})$	7
I5236		7
I5436		7
CØMT164	SLT XSR1	15
XSR1	(1F X59=1)	24
CØMT64		15
CEXPI3	X48-58 (EXP)	15
SEXPI3	UNBAS·SIGN EXT $\rightarrow$ I3 0·17	12
NCØ64		15
I3I2	I3 $\rightarrow$ I2	14
ENABF	ENABLE F RGTR	14
CØM64		
NINF1	SET NINF1 (IF EXP=3777+4000)	24
NIND1	SET NINF1 (IF EXP= 1777+6000)	24
NCØ50		
ENABLC	ENABLE C RGTR	7

TERM NAME	COMMAND OR FUNCTION	DPD NO
$\overline{I5S4} \cdot \overline{I5S2} \cdot \overline{I5S1}$	0'S NON COEF (I5 0-47, 96-107)	8
SH114	[X5R1]	22
SHI5C	COMP I5 96-107	22
SH114	[25]	22
SI5057	SET I5 2^{107}	22
SI5157	(I5S4·I5S2·I5S1)	22
SH114		22
SI5185	$C_{48-95} \rightarrow I5_{48-95}$ $(\overline{I5S4} \cdot I5S2 \cdot \overline{I5S1})$	22
SH114	[24+25]	22
ENABNN	ENABLE NORMALIZE	10
ENNØR	NETWORK	
EAEE	ENABLF EE RGTR	4
SH114		
SLNNI3	NN $\rightarrow$ I3	22
SH114	$[\overline{XSR1}]$	22
SHI3C	COMP I3	22
SHEC	NINF1 + NIND1	25

TERM NAME	COMMAND OR FUNCTION	DPD NO
SH164		
SLI9-0·		
SLI9-1	NN $\rightarrow$ I9	22
NSH164		
SKS1·SKS2	PRESET SK CNTR	9
SH164		
SHENBF	ENABLE F RGTR	22
SH164	24 + 25	
RS	SET LEFT SHIFT	22
SH164		
SI5185		22
SI5285	$X_{0-59} \rightarrow I5_{48-107}$	22
SI5164		22
SI5157	$(I5S4 \cdot I5S2 \cdot \overline{I5S1})$	22
SI5267		22
SN164	$[\overline{FFUNN4}]$	
BLI5C	COMP I5	21
SH164	24+25·SHEC	
SHENBC	ENABLE C RGTR	22

TERM NAME	COMMAND OR FUNCTION	DPD NO
SH264	$[*\overline{XKAECN}]$	
SI5085		22
SI5285	$SN_{48-95} \rightarrow I5_{48-95}$ $(I5S4 \cdot \overline{I5S2} \cdot I5S1)$	22
SH264	$F_{0-9} \rightarrow I15_{48-59}$	
FEXI15	F10, XSR1	8
SH264	$[*\overline{XKAECN}]$	22
SI5267	$I15_{48-59} \rightarrow I5_{96-107}$	8
SH264	$[*\overline{XKAECN}]$	
SHNEXT	ENABLE COMMON TIME, RNI	22
SH264	[24+25]	
SHI3I2	I3 $\rightarrow$ I2	22
ENABF	ENABLE F RGTR	13
SH264	$[**\overline{SHEC}]$	
SHENDC	ENABLE C RGTR	22
SH264	$[\overline{XKAEC}]$	
GØ264N	ENABLE END CHSE DELAY	22
	$*\overline{XKAECN}=$ $\left[\overline{UNDERFLOW} \cdot 24 \cdot \overline{COEFQO}\right]$ NINF1 + NIND1	
	$**\overline{SHEC}=$ $[\overline{NINF1} \cdot \overline{NIND1}]$	

TERM NAME	COMMAND OR FUNCTION	DPD NO
COMX00		15
SELXI	SELECT XI RGTR	2
SELBJ	SELECT BJ RGTR	2
COMENX	ENABLE WRITE	2
COMENB	STROBE X,B RGTR	
COMX00		5
HLSL	SELECT HIGHER	10
GØ264N		
SHENBC	ENABLE C RGTR	22
T264N1	ENABLE COMMON TIME, RNI	22

TERM NAME	COMMAND OR FUNCTION	DPD NO
COMX00		15
SELXI	SELECT XI RGTR	2
SELBJ	SELECT BJ RGTR	2
COMENX	ENABLE WRITF	2
COMENB	STROBE X, B RGTR	
COMX00		15
HLSL	SELECT HIGHER	10

CONTROL DATA
CANADIAN DEVELOPMENT DIVISION

INSTRUCTION FLOW SEQUENCE : SHIFT
INSTRUCTION : 24, 25

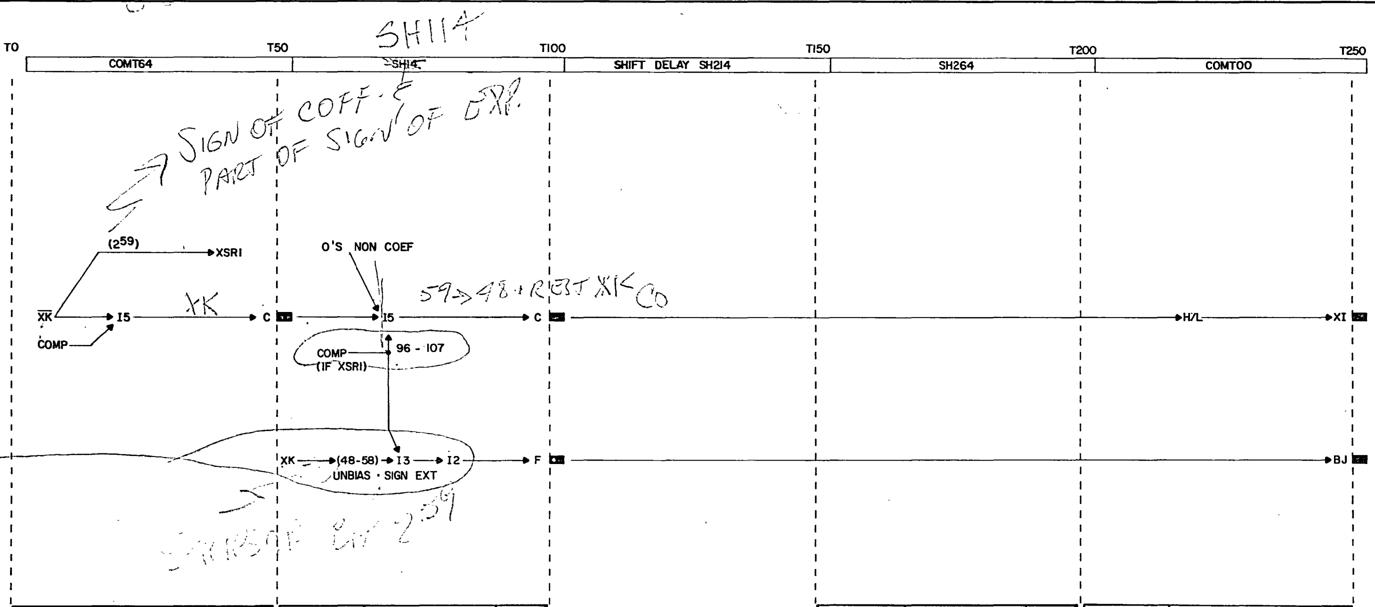
CODE IDENT		DWG. NO	REV
34570	D	19981800	A

FIGURE 5-2-17

sign of exponent
unbias = (+)
(−)
like =

15P 58 57
SIGN OF COFF
SIGN OF EXPONENT.

17



1. UNBIAS
2. COMPLEMENT
3. SIGN EXTEND.

TERM NAME	COMMAND OR FUNCTION	DPD NO
CØM50	FFE02	2
SELXK	SELECT XK RGTR	2
NC064		
I5185		7
I5285	$X_{0-59} \rightarrow I5_{48-107}$	7
I5164		7
I5264		7
I5236	(I5S4·I5S2·I5S1)	7
I5436		7
COM64		
I5C85	COMP $I5_{48-107}$	8
I5C67		8
COMT64	IFX59=1	
XSR1	SET XSR1	24
NCØ50		
ENABLC	ENABLE C RGTR	8

TERM NAME	COMMAND OR FUNCTION	DPD NO
$\overline{I5S4}\cdot\overline{I5S2}\cdot\overline{I5S1}$	O'S NON COEF (I5 0-47, 96-107)	8
SH114	[XSR1]	
SHI5C	COMP I5 96-107	22
SH100		
SHSLXK	SELECT XK RGTR	22
SH114		
SHEXI3	X48-58 (EXP)	22
SEXPI3	UNBIAS·SIGN EXT $\longrightarrow$ I3 0-17	12
SH114		
SHI3I2	I3 $\longrightarrow$ I2	22
ENABF	ENABLE F RGTR	13
SH114	[XSR1]	
SHI3C	COMP I3	22
SH114		
SI5185	$C_{48-95} \rightarrow I5_{48-95}$ ($\overline{I5S4}$·I5S2·$\overline{I5S1}$)	22

TERM NAME	COMMAND OR FUNCTION	DPD NO
SH264		
SHNEXT	ENABLE COMMON TIME, RNI	22

TERM NAME	COMMAND OR FUNCTION	DPD NO
COMX00		15
SELXI	SELECT XI RGTR	2
SELBJ	SELECT BJ RGTR	2
CØMENX	ENABLE WRITE	2
COMENB	STROBE X, B RGTR'S	2
$\overline{\text{COMX00}}$		15
HLSL	SELECT HIGHER	10

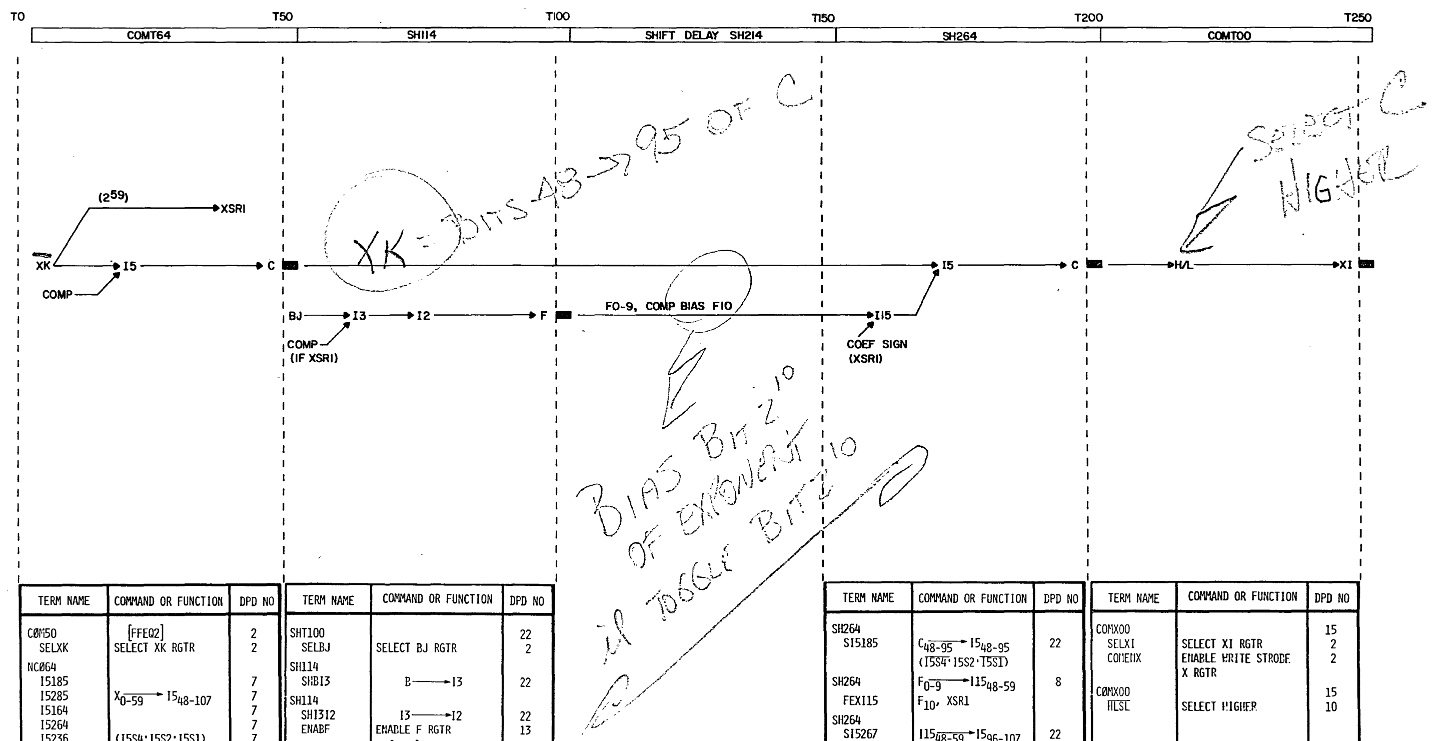
STUDENT MAN
HOTR 7.12

CONTROL DATA CANADIAN DEVELOPMENT DIVISION

INSTRUCTION FLOW
SEQUENCE : SHIFT
INSTRUCTION 26

CODE IDENT.		DWG. NO.	REV
34570	D	19981800	A
FIGURE 5-2-18		PAGE NO 5-2-46.6	

CARLETON INSTRUMENTS 138-3 200-12-74



TERM NAME	COMMAND OR FUNCTION	DPD NO
CØM50	[FFEQ2]	2
SELXK	SELECT XK RGTR	2
NCØ64		
I5185		7
I5285	$X_{0-59} \rightarrow I5_{48-107}$	7
I5164		7
I5264		7
I5236	(I5S4·I5S2·I5S1)	7
I5436		7
COM64		
I5C85	COMP I5 48-107	8
I5C67		8
CONT64	[IF X59=1]	
XSR1	SET XSR1	24
NCØ50		
ENABLC	ENABLE C RGTR	8

TERM NAME	COMMAND OR FUNCTION	DPD NO
SHT100		22
SELBJ	SELECT BJ RGTR	2
SH114		
SHBI3	$B \rightarrow I3$	22
SH114		
SHI3I2	$I3 \rightarrow I2$	22
ENABF	ENABLE F RGTR	13
SH114	[XSR1]	
SHI3C	COMP I3	23

TERM NAME	COMMAND OR FUNCTION	DPD NO
SH264		
SI5185	$C_{48-95} \rightarrow I5_{48-95}$ ($\overline{I5S4}$·I5S2·$\overline{I5S1}$)	22
SH264	$F_{0-9} \rightarrow I15_{48-59}$	8
FEXI15	F_{10}, XSR1	
SH264		
SI5267	$I15_{48-59} \rightarrow I5_{96-107}$ (I5S4·$\overline{I5S2}$·$\overline{I5S1}$)	22
SH264		
SHENBC	ENABLE C RGTR	22
SH264		
SHNEXT	ENABLE COMMON TIME, RNI	22

TERM NAME	COMMAND OR FUNCTION	DPD NO
COMX00		15
SELXI	SELECT XI RGTR	2
COMENX	ENABLE WRITE STROBE X RGTR	2
CØMX00		15
$\overline{HLSL}$	SELECT HIGHER	10

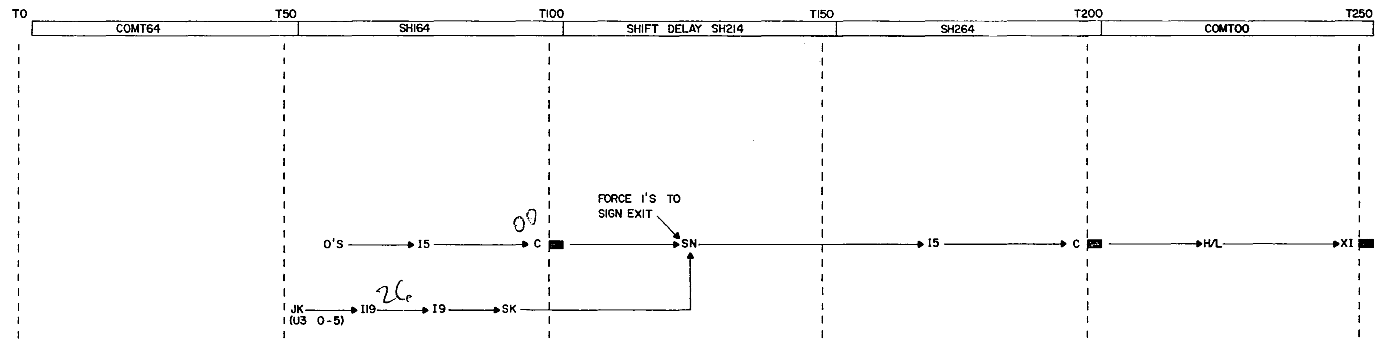
STUDENT MARK

CONTROL DATA CANADIAN DEVELOPMENT DIVISION	INSTRUCTION FLOW SEQUENCE: SHIFT INSTRUCTION: 27	CODE IDENT. 34570	D	DWG. NO. 19981800
		FIGURE 5-2-19		PAGE NO. 5-2-46-7

REV A

CARLETON INSTRUMENTS 138-3 300-12-74

HO TR-7.13



TERM NAME	COMMAND OR FUNCTION	DPD NO
$\overline{F46X}$	$U3_{0-5} \rightarrow I19$	9
SH164		22
$\overline{SLI90} \cdot \overline{SLI91}$	$I19 \rightarrow I9$	9
NSH164		22
SKS1·SKS2	$I9 \rightarrow SK$	9
$\overline{I5S4} \cdot \overline{I5S2} \cdot \overline{I5S1}$	$0'S \rightarrow I5$	8
SH164		
SHENBC	ENABLE C RGTR	22

TERM NAME	COMMAND OR FUNCTION	DPD NO
NF43	FORCE 1'S TO SIGN EXTENSION	10

TERM NAME	COMMAND OR FUNCTION	DPD NO
SH264		
SI5085		22
SI5285	$SN_{48-107} \rightarrow I5_{48-107}$	22
SI5064		22
SI5057	$(I5S4 \cdot \overline{I5S2} \cdot I5S1)$	22
SI5267		
SH264		
SHENBC	ENABLE C RGTR	22
SH264		
SHNEXT	ENABLE COMMON TIME, RNI	22

TERM NAME	COMMAND OR FUNCTION	DPD NO
CØMX00		15
SELXI	SELECT XI RGTR	2
COMENX	ENABLE WRITE STROBE X RGTR	2
CØMXOC		15
$\overline{HLSL}$	SELECT HIGHER	10

CONTROL DATA
CANADIAN DEVELOPMENT DIVISION

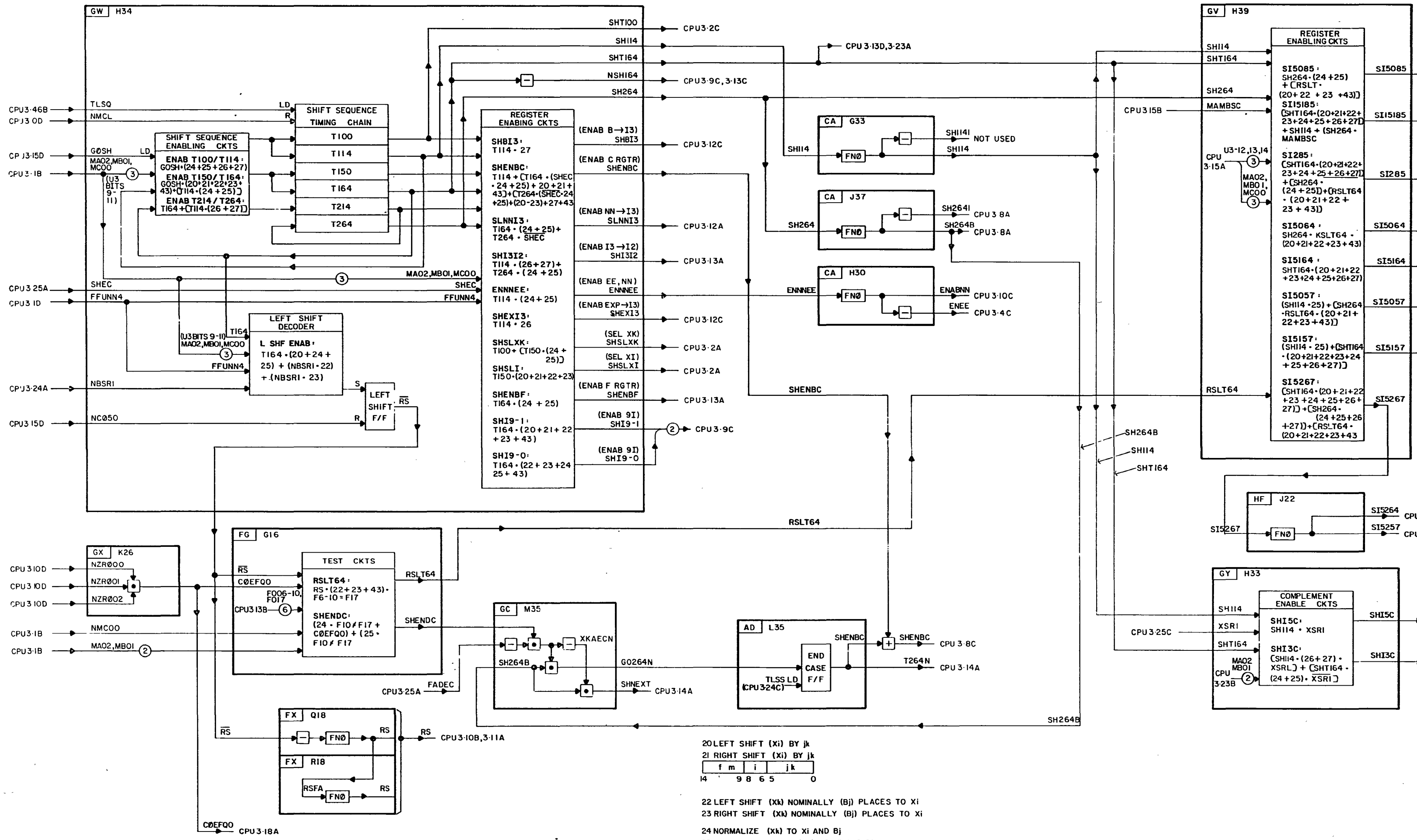
INSTRUCTION FLOW
SEQUENCE: SHIFT
INSTRUCTION: 43

CODE IDENT		DWG NO.	REV
34570	D	19981800	A

FIGURE 5-2-20

PAGE NO 5-2-46-8

CARLETON INSTRUMENTS 138-3 200-12-74



20 LEFT SHIFT (Xi) BY jk
21 RIGHT SHIFT (Xi) BY jk

f m	i	jk

14 9 8 6 5 0

22 LEFT SHIFT (Xk) NOMINALLY (Bj) PLACES TO Xi
23 RIGHT SHIFT (Xk) NOMINALLY (Bj) PLACES TO Xi

24 NORMALIZE (Xk) TO Xi AND Bj
25 ROUND NORMALIZE (Xk) TO Xi AND Bj

26 UNPACK (Xk) TO Xi AND Bj
27 PACK (Xk) AND (Bj) TO Xi

f m	i	j	k

14 9 8 6 5 3 2 0

CONTROL DATA CANADIAN DEVELOPMENT DIVISION

SHIFT SEQUENCE

CODE IDENT		DWG NO	REV
34570	D	19981800	L
PUB NO		SHEET CPU 3-22	PAGE NO 5-2-47

DETAILED PAK DIAGRAM (CPU 3.23)

BOOLEAN SEQUENCE

The Boolean sequence controls the operations necessary to perform the following instructions:

Transmit

10ijx	Transmit (Xj) to Xi
14ixk	Transmit the Complement of Xk to Xi

Logical

11ijk	Logical Product of (Xj) and (Xk) to Xi
12ijk	Logical Sum of (Xj) and (Xk) to Xi
13ijk	Logical Difference of (Xj) and (Xk) to Xi
15ijk	Logical Product of (Xj) and Complement (Xk) to Xi
16ijk	Logical Sum of (Xj) and Complement (Xk) to Xi
17ijk	Logical Difference of (Xj) and Complement (Xk) to Xi

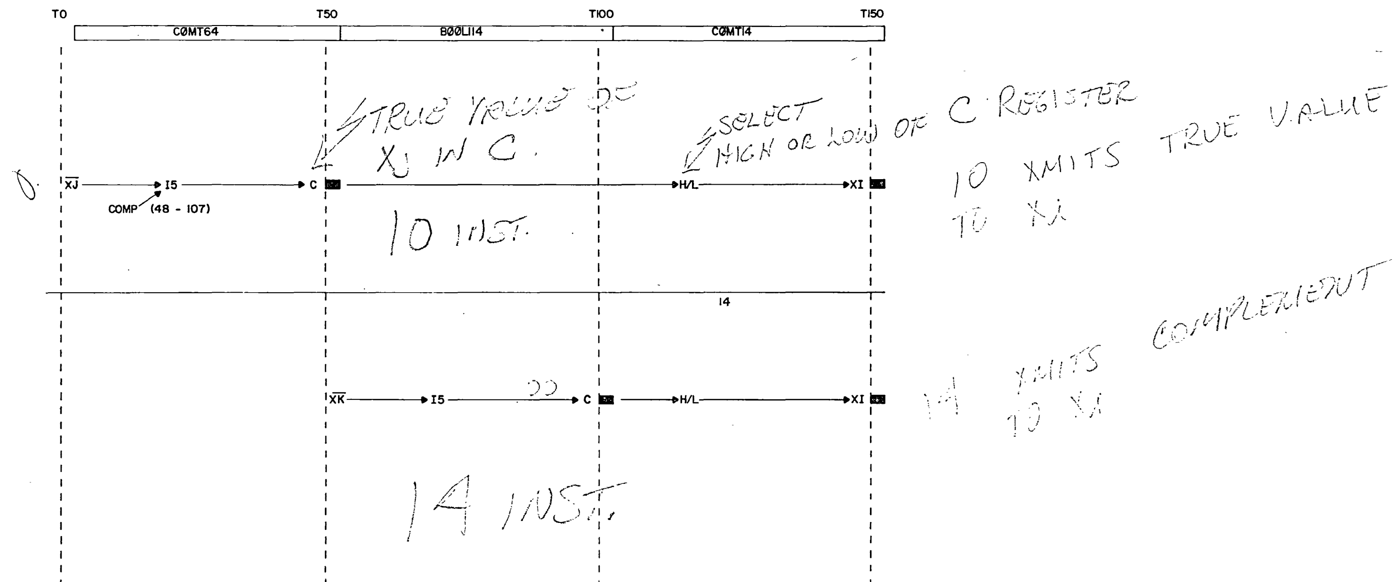
The 10 instruction transfers a 60-bit word from register Xj to register Xi.

The 14 instruction extracts the 60-bit word from operand register Xk, complements it, and transmits the complemented quantity to operand register Xi.

The 11-13 instructions perform the logical product (AND function), logical sum (inclusive OR function), and logical difference (exclusive OR function) of 60-bit words from operand registers Xj and Xk, and place the result in operand register Xi.

The 15-17 instructions perform the logical product (AND function), logical sum (inclusive OR function), and logical difference (exclusive OR function) of the 60-bit quantity from operand register Xj and the complement of the 60-bit word from operand register Xk, and place the result in operand register Xi.

The arithmetic operations for instructions 11-17 are performed by the D adder. The Boolean sequence controls the logical operation codes sent to the D adder which, in turn, directs the D adder ALU to perform the required logical operation.

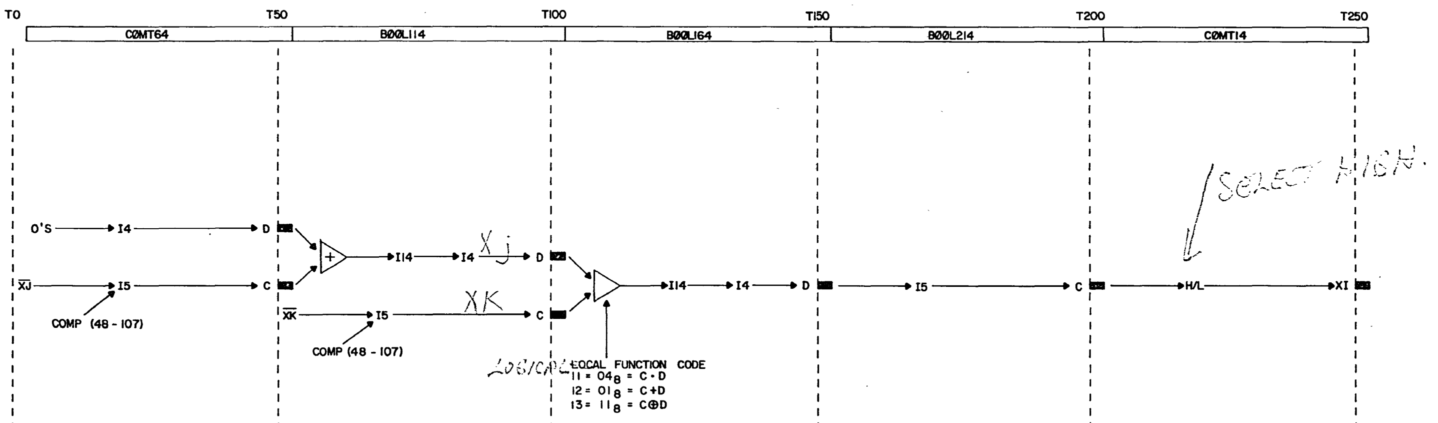


TERM NAME	COMMAND OR FUNCTION	DPD NO
CØM50		15
SELXJ	SELECT XJ RGTR	13
CØM64		15
I5C85	COMP I5 48-95	8
I5C67	COMP I5 96-107	8
NCØ64		15
I5185		7
I5285	$X_{0-59} \longrightarrow I5_{48-107}$	7
I5164		7
I5264		7
I5236	$(I5S4 \cdot I5S2 \cdot \overline{I5S1})$	7
I5436		7
CØM50		15
ENABLC	ENABLE C RGTR	8

TERM NAME	COMMAND OR FUNCTION	DPD NO
NB100		23
SXK		23
SELXK	SELECT XK RGTR	5
NB114		23
I5185		7
I5285	$X_{0-59} \longrightarrow I5_{48-107}$	7
I5164		7
I5264		7
I5236	$(I5S4 \cdot I5S2 \cdot \overline{I5S1})$	7
I5436		7
BØØL-T114	[14]	23
BLI5C	BLOCK COMP I5	
BØØL-T114		23
BØØENC	[14]	23
ENABC	ENABLE C RGTR	8
BØØL-T114	[10 + 14]	23
BØEXIT	ENAPLE RNI, COMMON TIME	23

TERM NAME	COMMAND OR FUNCTION	DPD NO
CONXOO		15
SELXI	SELECT XI RGTR	2
COMENX	ENABLE WRITE STROBE X RGTR	2
$\overline{\text{HLSL}}$	SELECT HIGHER	10

CONTROL DATA CANADIAN DEVELOPMENT DIVISION	INSTRUCTION FLOW SEQUENCE : BØØLEAN INSTRUCTION : 10, 14	CODE IDENT. 34570	D	DWG. NO. 19981800 REV A
		FIGURE 5 - 2 - 21		PAGE NO. 5-2-48·2



TERM NAME	COMMAND OR FUNCTION	DPD NO
CØM50		15
SELXJ	SELECT XJ RGTR	~~13~~ 2
CØM64		14
I5C85	COMP I5 48-95	8
I5C67	COMP I5 96-107	8
NCØ64		15
I5185		7
I5285	$X_{\overline{0-59}} \rightarrow I5_{48-107}$	7
I5164		7
I5264	$(I5S4 \cdot I5S2 \cdot \overline{I5S1})$	7
I5236		7
I5436		7
CØM50		15
ENABLC	ENABLE C RGTR	8
NC064		
I40·I41	O'S ⟶ I4	5
NCØ50		
ENABLD	ENABLE D RGTR	5

TERM NAME	COMMAND OR FUNCTION	DPD NO
BØØL-T114		23
BØØEND		23
ENABLD	ENABLE D RGTR	5
NB100		23
SXK		2
SELXK	SELECT XK RGTR	2
ND114		23
I5185		7
I5285	$X_{\overline{0-59}} \rightarrow I5_{48-107}$	7
I5164		7
I5264		7
I5236	$(I5S4 \cdot I5S2 \cdot \overline{I5S1})$	7
I5436		7
BØØL-T114	11 + 12 +13	
$\overline{BLI5C}$		23
I5C85	COMP I5 48-95	7
I5C67	COMP I5 96-107	7
BØØL-T114		23
BØØENC		23
ENABC	ENABLE C RGTR	8

TERM NAME	COMMAND OR FUNCTION	DPD NO
BØØL-T164		23
BOOEND		23
ENABLD	ENABLE D RGTR	5

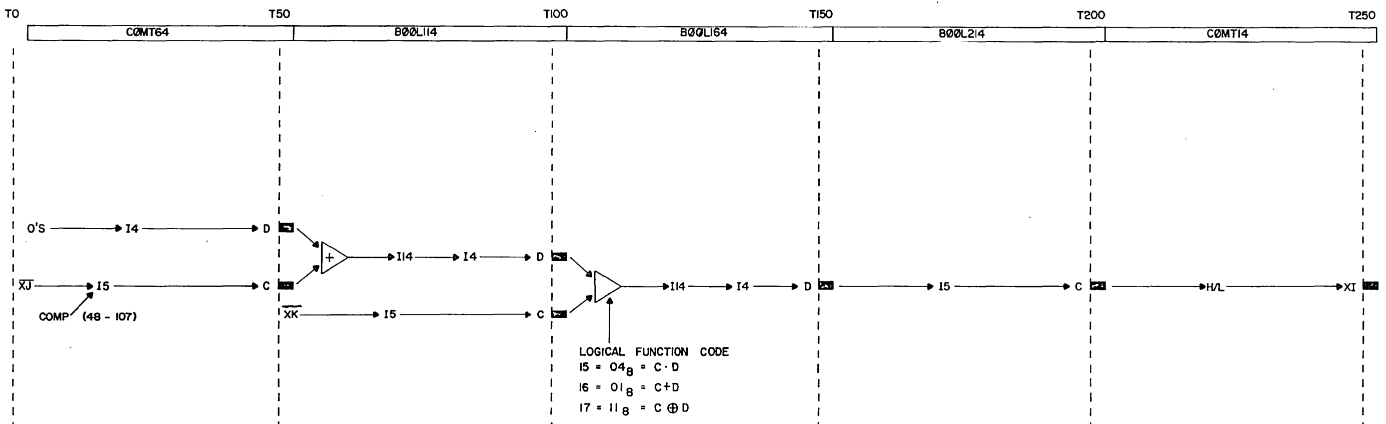
TERM NAME	COMMAND OR FUNCTION	DPD NO
ND214		23
I5085	$D_{\overline{48-107}} \rightarrow I5_{48-107}$	7
I5064		7
I5136	$(\overline{I5S4} \cdot \overline{I5S2} \cdot I5S1)$	7
BØØL-T214		23
BØØENC		23
ENABC	ENABLE C RGTR	8
BØØL-T214		23
BØEXIT	ENABLE RNI, COMMON TIME	23

TERM NAME	COMMAND OR FUNCTION	DPD NO
CØMX00		15
SELXI	SELECT XI RGTR	2
DOMENX	ENABLE WRITE STROBE X RGTR	2
$\overline{HLSL}$	SELECT HIGHER	10

CONTROL DATA
CANADIAN DEVELOPMENT DIVISION

INSTRUCTION FLOW
SEQUENCE : BOOLEAN
INSTRUCTION : 11, 12, 13

CODE IDENT.		DWG. NO.	REV
34570	D	19981800	A
FIGURE 5-2-22		PAGE NO. 5-2-48·3	



TERM NAME	COMMAND OR FUNCTION	DPD NO
CØN50		15
SELXJ	SELECT XJ RGTR	13
CØN164		15
I5C85	COMP I5 48-95	8
I5C67	COMP I5 96-107	8
NCØ64		15
I5185		7
I5285	$X_{0-59} \rightarrow I5_{48-107}$	7
I5164		7
I5264		7
I5236	$(I5S4 \cdot I5S2 \cdot \overline{I5S1})$	7
I5436		7
CØN50		15
ENABLD	ENABLE D RGTR	8
NCØ64		
I40·I41	$O'S \rightarrow I4$	5
NCØ50		
ENABLD	ENABLE D RGTR	5

TERM NAME	COMMAND OR FUNCTION	DPD NO
BØØL-T114		23
BØØEND		23
ENABLD	ENABLE D RGTR	5
NB100		23
SXK		2
SELXK	SELECT XK RGTR	2
NB114		23
I5185		7
I5285	$X_{0-59} \rightarrow I5_{48-107}$	7
I5164		7
I5264	$(I5S4 \cdot I5S2 \cdot \overline{I5S1})$	7
I5236		7
I5436		7
BØØL-T114	[I5 +I6 +I7]	23
BLI5C	BLOCK COMP I5	8

TERM NAME	COMMAND OR FUNCTION	DPD NO
BØØL-T164		23
BØØEND		23
ENABLD	ENABLE D RGTR	5

TERM NAME	COMMAND OR FUNCTION	DPD NO
NB214		23
I5085	$D_{48-107} \rightarrow I5_{48-107}$	7
I5064		7
I5136		7
BØØL-T214		23
BØØENC		23
ENABC	ENABLE C RGTR	8
BØØL-T214		23
BØEXIT	ENABLE RNI COMMON TIME	23

TERM NAME	COMMAND OR FUNCTION	DPD NO
COMX00		15
SELXI	SELECT XI RGTR	2
COMENX	ENABLE WRITE STROBE X RGTR	2
$\overline{HLSL}$	SELECT HIGHER	10

CONTROL DATA CANADIAN DEVELOPMENT DIVISION

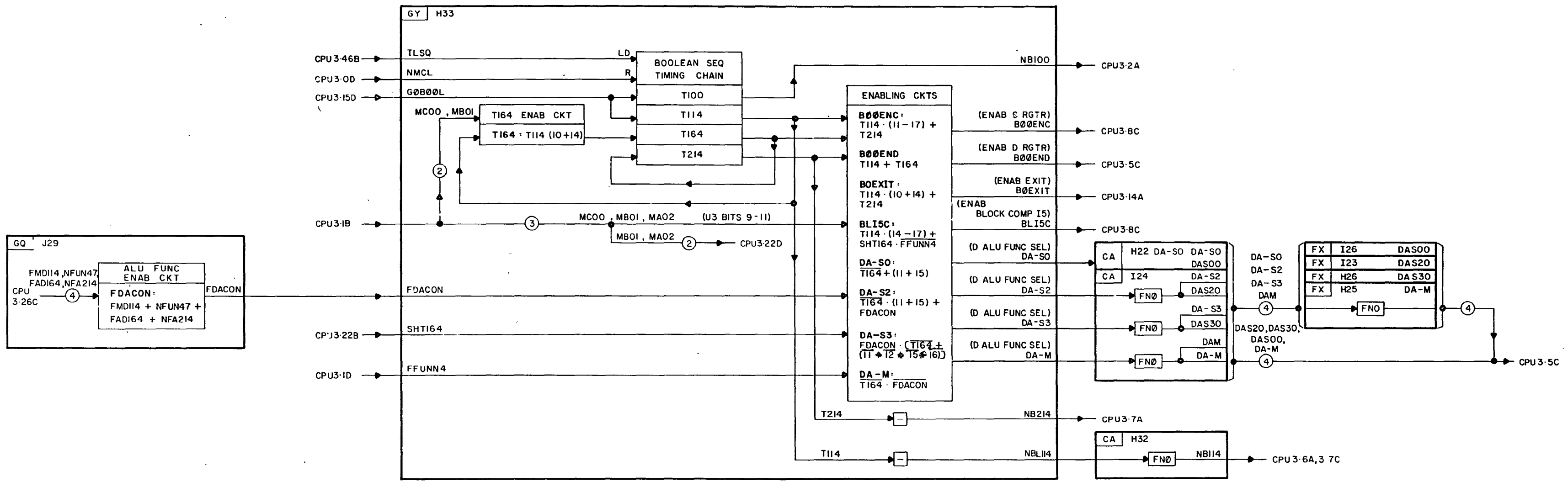
INSTRUCTION FLOW
SEQUENCE: BOOLEAN
INSTRUCTION: 15, 16, 17

CODE IDENT		DWG. NO.	REV
34570	D	19981800	A

FIGURE 5-2-23

PAGE NO. 5-2-48-4

CARLETON INSTRUMENTS 128-3 200-12-74



10 ijk TRANSMIT (Xj) TO Xi

fm	i	j	

14 9 8 6 5 3 2 0

11 ijk LOGICAL PRODUCT OF (Xj) AND (Xk) TO Xi
12 ijk LOGICAL SUM OF (Xj) AND (Xk) TO Xi
13 ijk LOGICAL DIFFERENCE OF (Xj) AND (Xk) TO Xi

fm	i	j	k

14 9 8 6 5 3 2 0

14 ixk TRANSMIT COMPLEMENT OF (Xk) TO Xi

fm	i	j	k

14 9 8 6 5 3 2 0

15 ijk LOGICAL PRODUCT OF (Xj) AND COMPLEMENT OF (Xk) TO Xi
16 ijk LOGICAL SUM OF (Xj) AND COMPLEMENT OF (Xk) TO Xi
17 ijk LOGICAL DIFFERENCE OF (Xj) AND COMPLEMENT OF (Xk) TO Xi

fm	i	j	k

14 9 8 6 5 3 2 0

CONTROL DATA CANADIAN DEVELOPMENT DIVISION	BOOLEAN SEQUENECE	CODE IDENT. 34570	D	DWG NO. 19981800	REV A
		PUB NO		SHEET CPU3·23	PAGE NO. 5-2-49

Q ⇒ ROUND BIT

DETAILED PAK DIAGRAM (CPU 3.24, 3.25, 3.26)

FLOATING POINT ADD SEQUENCE (FAD)

The FAD sequence controls the operations necessary to perform the sum or difference of two floating point quantities in Xj and Xk. The packed result is returned to the Xi register.

The floating point instructions controlled by the FAD sequence are as follows:

30ijk	Floating Sum of (Xj) and (Xk) to Xi
31ijk	Floating Difference of (Xj) and (Xk) to Xi
32ijk	Floating Double Precision Sum of (Xj) and (Xk) to Xi
33ijk	Floating Double Precision Difference of (Xj) and (Xk) to Xi
34ijk	Round Floating Sum of (Xj) and (Xk) to Xi
35ijk	Round Floating Difference of (Xj) and (Xk) to Xi

The FAD sequence is initiated by GOFAD from the common time sequence. The operands are obtained from the selected Xj and Xk registers. The exponents are extracted and tested for infinite ($3777_8 + 4000_8$) or indefinite ($1777_8 + 6000_8$) operands. An infinite or indefinite operand causes the FAD sequence to abort and enables the end case exit sequence.

The floating sum or difference operation involves the addition of two floating point coefficients that have equal exponents. Exponent equalization is accomplished by right shifting the coefficient of the smaller exponent a number of places equal to the absolute difference of the two exponents. A right shift decreases the size of the coefficient (moves the binary point left) and the exponent is therefore made larger. Once the exponents are equalized, the sum or difference of the coefficients is computed in the D adder. At the conclusion of the add operation, the binary point is considered to be located between bit positions 47 and 48 of the 108-bit D register.

Single precision instructions (30, 31, 34, 35) use the coefficient result contained in bit positions 48-95 of the D register, and pack the computed exponent. Double precision instructions (32,33) use the lower 48 bits of the D register and subtract 60_8 from the computed exponent before packing. This shifts the binary point to the right of bit 0 which is necessary to express the result as an integer.

COEFFICIENT LARGER THEREFORE EXPONENT MUST BE MADE SMALLER.

Coefficient overflow is checked during FAD364 by examining D register bits 96 and 97. If D register bits 96 $\neq$ 97, coefficient overflow has occurred. The coefficient is right shifted by one, and the exponent is increased by one.

Exponent underflow is checked during FAD414. Underflow is detected when the exponent is less than -1777_8 after correction during FAD364. Exponent underflow causes the FAD sequence to abort normal exit and enables the end case exit sequence.

The final coefficient and exponent plus bias are packed in I5 during FAD414. D register bit 107 controls complementing the exponent if the resulting coefficient sign is negative. FAD414 enables the common time sequence (COMT00) and the RNI sequence. Common time allows the contents of C to be stored in Xi.

ROUND OPERATION (34, 35 INSTRUCTIONS)

The 34 and 35 instructions operate in the same manner as described, except that the coefficients are rounded before the addition process to produce a rounded sum or difference.

The round bit is attached at the right end of both coefficients (bit 47) during FAD114 and FAD164. During FAD214, the round bit is removed from the coefficient with the smaller exponent when the following conditions are present:

1. $34 \cdot \overline{BON} \cdot XSR1 = XSR2$; or } BOTH OPERANDS NOT NORMALIZED.
2. $35 \cdot \overline{BON} \cdot XSR1 \neq XSR2$

The round bit increases the absolute value of the coefficient by one half the value of the least significant bit.

FLOATING POINT MULTIPLY/DIVIDE SEQUENCE (FMD)

The FMD sequence controls the operations necessary to perform multiplication or division of floating point quantities in Xj and Xk. Multiply instructions 40, 41, 42, form the product of multiplier Xj times multiplicand Xk and send the result to Xi. Divide instructions 44 and 45, form the quotient of the dividend Xj divided by the divisor Xk and send the result to Xi.

The FMD sequence also controls the operations necessary to count the number of one-bits in Xk (population count instruction 47) and store the result in Xi.

The floating multiply and divide instructions controlled by the FMD sequence are as follows:

40ijk	Floating Product of (Xj) and (Xk) to Xi
41ijk	Round Floating Product of (Xj) and (Xk) to Xi
42ijk	Floating Double Precision Product of (Xj) and (Xk) to Xi
44ijk	Floating Divide (Xj) by (Xk) to Xi
45ijk	Round Floating Divide (Xj) by (Xk) to Xi

DIVIDE SUBTRACTS EXPONENTS.

PREPARATION OF OPERANDS

The operands are obtained from the selected Xj and Xk registers. The exponents are extracted and tested for infinite ($3777_8 + 4000_8$), indefinite ($1777_8 + 6000_8$), or zero ($0000_8 + 7777_8$). An infinite, indefinite or zero operand causes the FMD sequence to abort and enables the end case exit sequence. Zero exponents in both Xj and Xk enable integer multiply. Integer multiply blocks end case exit. ZERO EXPONENTS = INTERGER

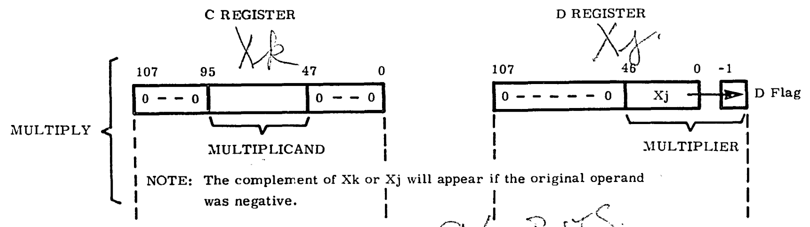
The bias for each exponent is removed in I3 and sign extended. The Xj exponent is transferred from I3 through I2 to F. The Xk exponent is transferred from I3 to E. With both exponents at the input to the small adder, a subsequent add during FMD2714 produces the final result exponent before any correction is made.

MULTIPLY COEFFICIENTS & ADD THE EXPONENTS.

MULTIPLY STEPS

During common time, the shift and iteration counter is preset with 60_8 to allow the Xj coefficient to be shifted right 48 bits to align with bit 0 of the C register. Since all numbers are considered integers rather than fractions, the binary point is considered as being to the right of bit 0. Right shifting the Xj coefficient (multiplier) 48 bits places it in the proper position for the multiplication process.

The C and D registers initially appear at the input of the D adder as follows:



PRODUCT = 96 BITS.

Just before the multiply iterations, the Xj coefficient is transferred via I14 to I4 where a right shift of one occurs. The shifted bit is sent to the D flag register. The multiply iterations are performed during FMD264 through FMD2664. The SK counter contains the 60_8 iteration count. Each 50 ns clock pulse decrements the counter by one until all iterations have been performed.

The D flag monitors the condition of the lowest order bit of D. Before the first iteration, the multiplier was right shifted one into the D flag The D flag now determines the first operation. If the D flag is set, the output of the D adder is right shifted one and sent back to the D register. If the D flag is clear, the output of the D register is right shifted one and sent back to the D register. After the first iteration, the D register holds the partial product and the remaining bits of the multiplier. This process continues until the quantity in SK is reduced to zero. After the last iteration, the D register contains the final product with the multiplier shifted end off out of the register.

On the last iteration, bit 46 of C is set while the rest of C is cleared to zeros. C is added to the product in D during FM2714 to form a rounded result. The rounded product is sent to the D register on a 41 instruction only.

EXPONENT AND RESULT FORMATION, MULTIPLY

The final exponent for the 96-bit product is formed in the F adder during FM2714. For single precision instructions 40, 41, the exponent would already have been adjusted by 60_8. Adjustment of the exponent for single precision instructions is performed during FMD164, where 60_8 is added to the Xj exponent in F. The exponent is therefore made relative to the upper 48 bits of the product, or zero is added to maintain an exponent relative to the 96-bit product.

If it is necessary to normalize the product during FM2764, the quantity 1 is subtracted from the result exponent in F, while the product in D is left shifted by one through I4 and returned to D.

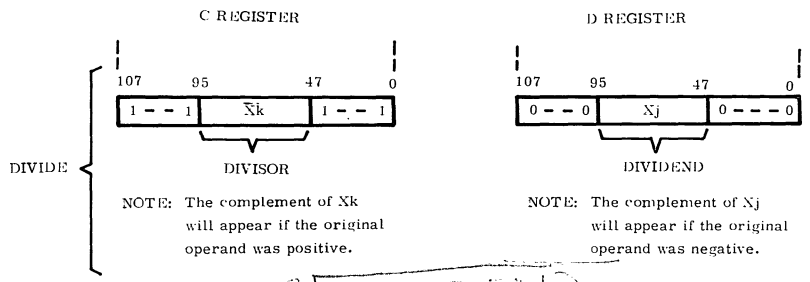
Exponent overflow or underflow is checked during FM2814 by determining that the absolute value of the exponent is greater than 1777_8. Exponent overflow or underflow causes the FMD sequence to abort normal exit, and enable the end case exit sequence.

The final product and exponent plus bias are packed in I5 during FM2814. I5 is complemented if XSR1 ≠ XSR2. FM2814 enables the common time sequence (COMT00) and the RNI sequence. Common time allows the upper or lower product from C, plus exponent and signs, to be stored in Xi.

✓

DIVIDE STEPS

Division is accomplished by repetitive subtractions in the D adder. The D register contains the coefficient of the dividend Xj and the C register contains the complemented coefficient of the divisor Xk. The C and D registers initially appear at the input to the D adder as follows:


C REGISTER
D REGISTER
107
95
47
0
1 - - 1
X̄k
1 - - 1
0 - - 0
Xj
0 - - - 0
DIVISOR
DIVIDEND
DIVIDE
NOTE: The complement of Xk will appear if the original operand was positive.
NOTE: The complement of Xj will appear if the original operand was negative.

DIVISOR⟌DIVIDEND

Before the first divide iteration, the Xj coefficient (dividend) in the C register is transferred via I14 to I4 where a right shift of one occurs. This reduces the dividend by one half. The dividend now in D is subtracted from the divisor (Xk coefficient) in C. If an end-around-carry occurs as a result of the subtraction, a divide fault is detected, since the coefficient of the dividend must be less than twice that of the divisor. A divide fault aborts the FMD sequence and enables end case exit.

The divide iterations are performed during FMD264 through FMD2664. The SK counter contains the 60_8 iteration count. Each 50 ns clock pulse decrements the counter by one until all iterations have been performed.

Each iteration checks for an end around carry condition from the D adder after the divisor in C has been subtracted from the dividend in D. If end around carry does not occur, the dividend in D is left shifted one place through I4 and returned to D before the next iteration. If end around carry does occur, a quantity one is gated to I14 bit position 0 and the D adder output is left shifted one place through I4 and sent to D. In this way the D register receives an additional quotient bit for each iterative step as the dividend is left shifted through the register. This process continues until the quantity in SK is reduced to zero. After the last iteration, the D register will contain the complete quotient in the lower 48 bits and the remainder in bit positions 48 through 95.

At this time the binary point is considered to be between bit positions 46 and 47 and must be shifted to the right of bit 0 to represent the quotient as an integer. This is accomplished by subtracting 57_8 from the Xj exponent during FMD114.

ROUND OPERATION

Rounding is accomplished by adding a quantity of 1/3 during the division process. Round bits are added during the divide steps of a 45 instruction each time the SK register contains an even count, except during the first iteration divide. This forces a 1-bit into the D register bit 48 so that successive iterations bring in the 1/3 round quantity of 25 --------------25_8.

EXPONENT AND RESULT FORMATION, DIVIDE

The final exponent for the quotient is formed by subtracting the exponent of the divisor Xk from the exponent of the dividend Xj in the F adder during FMD2764. The exponent of the dividend Xj will already have had a constant of 57_8 subtracted from the exponent value. The result exponent formed in the F adder will thus represent the coefficient as an integer.

During FM2764, the quotient is checked for normalization (D register bit 47 $\neq$ 0). If it is necessary to normalize the quotient, the quantity 1 is subtracted from the result exponent in F while the D register is shifted left by one through I4. If the remainder in D between bit positions 48-95 is $\geq$ the divisor in C, an end around carry from the D adder sets bit 0 of the quotient through I4. A normalized result is thus formed and returned to D.

Exponent overflow or underflow is checked during FM2814 by determining that the absolute value of the exponent is greater than 1777_8. Exponent overflow or underflow causes the FMD sequence to abort normal exit, and enable the end case exit sequence.

The final quotient and exponent plus bias are packed in I5 during FM2814. I5 is complemented if XSR1 $\neq$ XSR2. FM2814 enables the common time sequence (COMT00) and the RNI sequence. Common time allows the quotient from C, plus exponent and signs, to be stored in Xi.

END CASE SEQUENCE

The end case sequence checks the formation of infinite, indefinite and zero results when executing floating point instructions controlled by the FAD and FMD sequences.

[illegible]

[illegible]

The FAD sequence enables the end case sequence at FAD214 time when an infinite or indefinite operand is detected, or at FAD414 time when underflow is detected.

The FMD sequence enables the end case sequence at FMD214 time when an infinite, indefinite or zero operand is detected (except during multiply when both operands are zero); when a divide fault is detected; or, at FMD2814 time, when overflow or underflow is detected.

Overflow and Underflow

Exponents lying outside the range -1777_8 to $+1777_8$ cannot be generated during execution of floating point arithmetic instructions. An attempt to generate an exponent greater than $+1777_8$ yields an infinite result (overflow). An attempt to generate an exponent less than -1777_8 yields a zero result (underflow).

Indefinite

A positive indefinite (1777_8) or negative indefinite (6000_8) operand generates an indefinite indicator plus zero coefficient to the C register. A positive indefinite result indicator is generated whenever a calculation cannot be resolved. The indefinite indicator corresponds to a -0 exponent and a zero coefficient.

The common time and RNI sequences are enabled by the end case sequence after the proper 60-bit result is sent to the C register. Common time allows the end case result in C to be stored in the Xi register.

ERROR EXIT CONDITIONS

If an attempt is made to use an indefinite or infinite operand in floating arithmetic sequences, an optional exit mode selection is provided. The CPU response is dependent on whether the appropriate exit mode selection was made and the monitor flag /MEJ/CEJ condition.

An exit condition sensed (ECONDS) sets the error exit FF (CPU 3.17) at the same time as the next RNI sequence is initiated. Error exit clears the U3 instruction register, thus forcing a return jump error exit sequence.

POPULATION COUNT 47

The population count instruction is controlled by the FMD sequence. The instruction counts the number of 1-bits from a selected Xk register and delivers the count value to a selected Xi register.

TABLE 5-2-17. OVERFLOW AND UNDERFLOW CONDITIONS

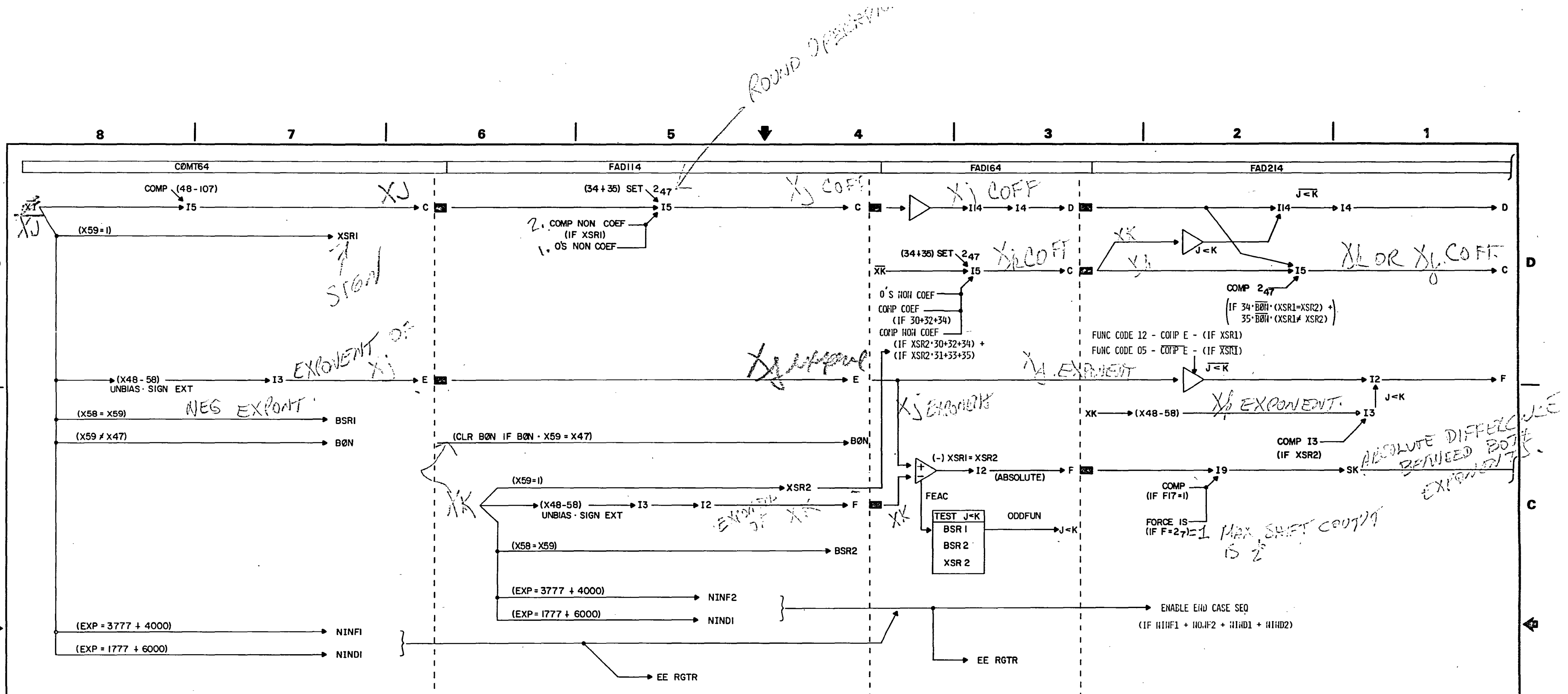
OVERFLOW		
INSTRUCTIONS	OVERFLOW CONDITION	RESULT
Upper Sum (30, 31, 34, 35)	None (see Note below.)	---
Lower Sum (32, 33)	None	---
Upper Product (40, 41)	$*n_1 + n_2 + 60_8 \geq 2000_8$	$X_i = 3777\ 0....0_8$ or $4000\ 0....0_8$ (True Sign)
Lower Product (42)	$n_1 + n_2 \geq 2000_8$	
Quotient	$n_1 - n_2 - 57_8 \geq 2000_8$	
UNDERFLOW		
INSTRUCTIONS	UNDERFLOW CONDITION	RESULT
Upper Sum (30, 31, 34, 35)	None	---
Lower Sum (32, 33)	Final Exponent $\leq -2000_8$	$X_i = 0000\ 0....0_8$
Upper Product (40, 41)	$n_1 + n_2 + 57_8 \leq -2000_8$	
Lower Product (42)	$n_1 + n_2 - 1 \leq -2000_8$	$X_i = 0000\ 0....0_8$
Quotient (44, 45)	$n_1 - n_2 - 60_8 \leq -2000_8$	

* n_1 and n_2 are the initial exponents.

Note: Overflow of Upper Sum: Overflow cannot occur unless one operand is infinite. In this case the result is as indicated. If a one-place Right Shift occurs when the larger operand exponent is equal to $+1776_8$, a correct result with exponent $+1777_8$ is generated.

The counting process is accomplished by left shifting the Xk operand in the D register one bit at a time into the D flag register. For every 1-bit shifted into the D flag, +1 is gated to the F register. The SK counter contains a count of 74_8, providing the required iterations to shift each bit into the D flag.

The resulting count value (maximum 74_8) is gated from F to the C register, and during common time to the selected Xi register.



TERM NAME	COMMAND OR FUNCTION	DPD NO
CØM50	SELECT XJ RGTR	15
SELXJ		2
CØM64		15
I5C85	COMP I5 48-95	8
I5C67	COMP I5 96-107	8
CØM50		15
ENABLC	ENABLE C RGTR	8
CØMT64		15
XSR1	SET XSR1 (IF X59=1)	24
CØMT64	X48-58 (EXP)	15
CEXPI3	UNPIAS SIGN EXT	15
SEXPI3	$\longrightarrow$ I3 0-17	12
NCØ50		15
ENABE	ENABLE E RGTR	13
CØM64		15
BSR1	SET BSR1 (IF X58= X59	24
CØM64		
NINF1	SET NINF1 (IF EXP= 3777+4000)	24
NIND1	SET NIND1 (IF EXP= 1777+6000)	24
COM64		
BON	SET BON FF (IF X59= X47)	24

NCØ64		
I5185	$X_{\overline{0-59}} \longrightarrow I5_{48-107}$	7
I5285		7
I5164		7
I5264	$(I5S4 \cdot I5S2 \cdot \overline{I5S1})$	7
I5236		7
I5436		7

TERM NAME	COMMAND OR FUNCTION	DPD NO
$\overline{I5S1} \cdot \overline{I5S2} \cdot \overline{I5S4}$	0'S $\longrightarrow$ NON COEF (I5 0-47, 96-107)	8
FD114·X5R1		25
FI5C06	COMP I5 0-46	25
FI5C47	COMP I5 47	25
FI5C67	COMP I5 96-107	25
FD114	34+35	
FI5057	1 $\longrightarrow$ I5 47	25
FI5157	$(\overline{I5S4} \cdot I5S2 \cdot I5S1)$	
FAD100		24
FASXK	SELECT XK RGTR	24
FD114	X48-58 (EXP)	24
FEXPI3	UNBIAS SIGN EXT	26
SEXPI3	$\longrightarrow$ I3 0-17	12
FD114		24
ENXSR2		26
BSR2	SET BSR2 (IF X58= X59)	
NINF2	SET NINF2 (IF EXP= 3777+4000)	24
NIND2	SET NIND2 (IF EXP= 1777+6000)	24
XSR2	SET XSR2 (IF X59=1)	24
FD114		24
FI3I2	I3 $\longrightarrow$ I2	26
ENABF	ENABLE F RGTR	13
FD114		
FENBC	ENABLE C RGTR	25

FAD114		
BØN	CLR BØN FF (IF BON·X59=X47)	24
FD114		
FENBEE	ENABLE EE RGTR	26
FD114		
FI5185	$C_{\overline{48-95}} \longrightarrow I5_{48-95}$ $(I5S4 \cdot I5S2 \cdot \overline{I5S1})$	25

TERM NAME	COMMAND OR FUNCTION	DPD NO
FAD150		24
FASXK	SELECT XK RGTR	24
$\overline{FDI14}$	D-ADD $\longrightarrow$ I14	26
FD164		
FI5C06	$[XSR2 \cdot 30+32+34]$	25
FI5C47	$+ [\overline{XSR2} \cdot 31+33+35]$	25
FI5C67	COMP I5 0-47,96-107	25
FD164		
FI5C85	30+32+34 COMP I5 48-95	25
$\overline{I5S1} \cdot \overline{I5S2} \cdot \overline{I5S4}$	0'S NON COEF I5 0-47, 96-107	
FD164		
FI5057	[34+35]	25
FI5157	$(\overline{I5S4} \cdot I5S2 \cdot I5S1)$	25
FD164		
FENBC	ENABLE C RGTR	25
FD164		
FENBD	ENABLE D RGTR	26
FD164		
FENBF	ENABLE F RGTR	26
FD164		
FI5185	$X_{\overline{0-47}} \longrightarrow I5_{48-95}$	25
FI5285	$(I5S4 \cdot I5S2 \cdot \overline{I5S1})$	25
FD164		
FENBEE	ENABLE EE RGTR	26

TERM NAME	COMMAND OR FUNCTION	DPD NO
FD214·$\overline{JLTR}$	$(\overline{J<K})$	24
FDI14	D $\longrightarrow$ I14	26
FDI14	C $\longrightarrow$ D-ADD $\longrightarrow$ I14	26
FD214·JLTK	(J<K)	
FI5004		25
FI5057	$D_{\overline{0-17}} \longrightarrow I5_{0-107}$	25
FI5085		25
FI5067	$(\overline{I5S4} \cdot \overline{I5S2} \cdot I5S1)$	25
FD214·JLTK	$(\overline{J\ K})$	
FI5104		25
FI5157	$C_{\overline{0-107}} \longrightarrow I5_{0-107}$	25
FI5185		25
FI5167	$(\overline{I5S4} \cdot I5S2 \cdot \overline{I5S1})$	25
FD214		
FENBD	ENABLE D RGTR	26
FENBC	ENABLE C RGTR	25
NFA214·$\overline{XSR1}$	E $\longrightarrow$ $\overline{FADD}$ $\longrightarrow$ I2 (XJ UNBIASED EXP)	25
FAD200		
FASXK	SELECT XK RGTR	24
NFA214	X48-58 (EXP)	24
FEXPI3	UNBIAS SIGN EXT	26
SEXPI3	$\longrightarrow$ I3 0-17	12

FD214·JLTK	(J K)	24
FI3I2	I3 $\longrightarrow$ I2 $\longrightarrow$ F	26
ENABF	ENABLE F RGTR	13
FD214·$\overline{JLTK}$	$(\overline{J<K})$	
FENBF	I2 $\longrightarrow$ F	26
FD214		
FENSK	ENABLE PRESENT SK	26
SLI91·SLI90·$\overline{FCOM}$	F $\longrightarrow$ I9	9
SLI91·SLI90·FCOM	F $\longrightarrow$ I9	9
FL128·$\overline{2X}$	1'S $\longrightarrow$ I9	9
F0214		
FAENDC	ENABLE END CASE SEQ	24

(Part 1 of 2)

CONTROL DATA CANADIAN DEVELOPMENT DIVISION

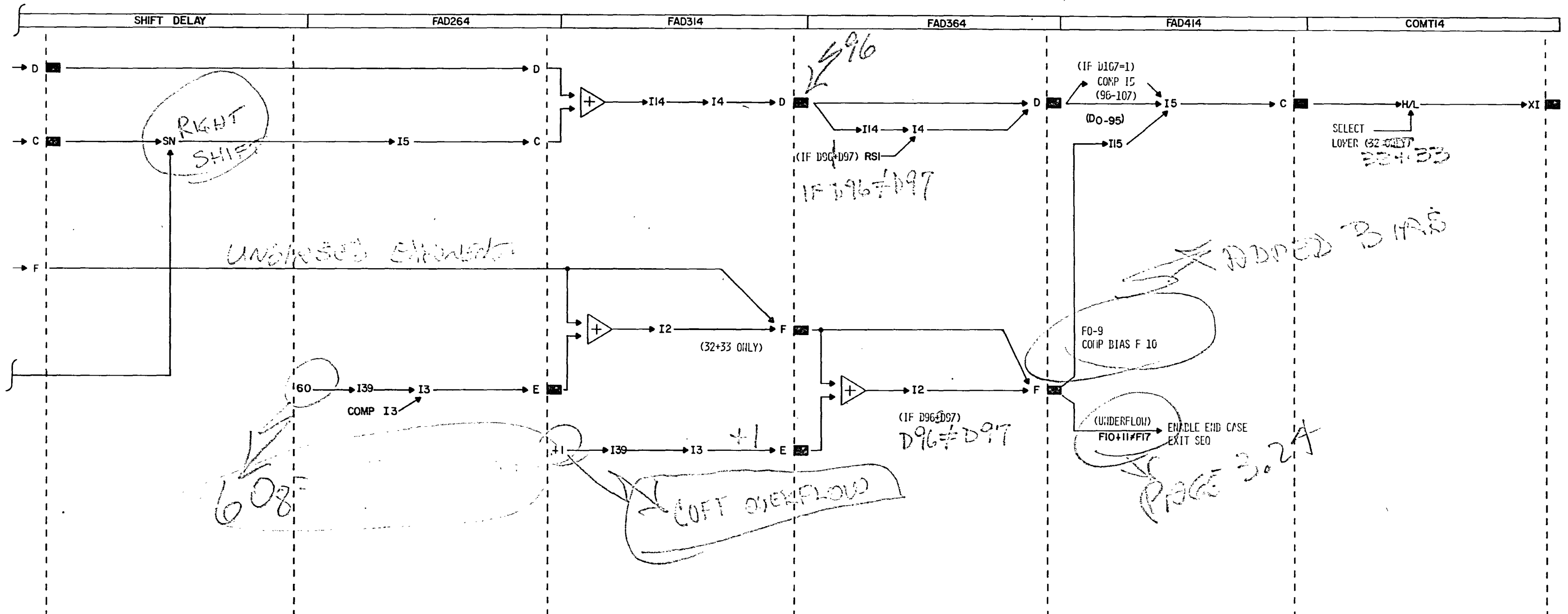
INSTRUCTION FLOW
SEQUENCE : FAD
INSTRUCTION : 30,32,31,33,34,35

CODE IDENT.		DWG. NO.	REV
34570	D	19981800	A

FIGURE 5-2-24

PAGE NO. 5-2-50-4

CARLETON INSTRUMENTS 138-3 200-12-74



TERM NAME	COMMAND OR FUNCTION	DPD NO
NOTE:	COMMANDS ARE SHOWN FOR FAD214	
F0214		
FI5C47	[34·$\overline{BON}$·XSR1=XSR2 +35·$\overline{BON}$·XSR1=XSR2] COMP I5 47 ($\overline{I5C1}$·I5C2)	25
FD214		
FAMI3C	[XSR2]	26
SCOMI3	COMP I3	12

TERM NAME	COMMAND OR FUNCTION	DPD NO
FAD264		
FI5004		25
FI5204	$SN_{0-107} \rightarrow I5_{0-107}$	25
FI5057		25
FI5257		25
FI5085	(I5S4·I5S2·I5S1)	25
FI5285		25
FI5067		25
FI5267		25
FD267		
FENBC	ENABLE C RGTR	25
FD264		
CONST4	$60_8 \rightarrow I39$	1
CONST5		
FD264		
FI39I3	I39 → I3	26
SI39I3		12
FD264		
FENBE	ENABLE E RGTR	26
FD264		
FAMI3C	COMP I3	26
SCOMI3		12

TERM NAME	COMMAND OR FUNCTION	DPD NO
$\overline{FDI14}$	D-ADD → I14	26
FD314		
FENBD	ENABLE D RGTR	26
FD314		
CONSTO	+1 → I39	1
FD314		
FI39I	I39 → I3	26
SI39I3		12
FD314		
FENBE	ENABLE E RGTR	26
FD314	[32+33]	
FENBF	ENABLE F RGTR	26

TERM NAME	COMMAND OR FUNCTION	DPD NO
FAD364·		
D96N97	I14 → I4	
I14-21·$\overline{I4-20}$	RS1 →	26
FAD364		
FDI14	D → I14	26
FAD364		
FENBD	ENABLE D RGTR	26
FAD364·	D96≠97	
D96N97		
FENBF	ENABLE F RGTR	26

TERM NAME	COMMAND OR FUNCTION	DPD NO
FAD414		
FI5004	$D_{0-95} \rightarrow I5_{0-95}$	25
FI5057		25
FI5085	($\overline{I5S4}$·$\overline{I5S2}$·I5S1)	25
FI5267		25
FEXI5	$F\ 0-9 \rightarrow I15_{48-59}$ $\overline{F10}$	8
FI5267	$I15_{48-59} \rightarrow I5_{96-107}$	25
FD414·D107		
FI5C67	COMP $I5_{96-107}$	25
FD414		
FENBC	ENABLE C RGTR	25
FD414		
FADEXT	ENABLE RNI COMMON TIME	24

TERM NAME	COMMAND OR FUNCTION	DPD NO
COMX00		15
SELXI	SELECT XI RGTR	2
COMENX	ENABLE WRITE STROBE X RGTR	2
COMX00	[32+33]	15
HLSL	SELECT LOWER	10

(Part 2 of 2)

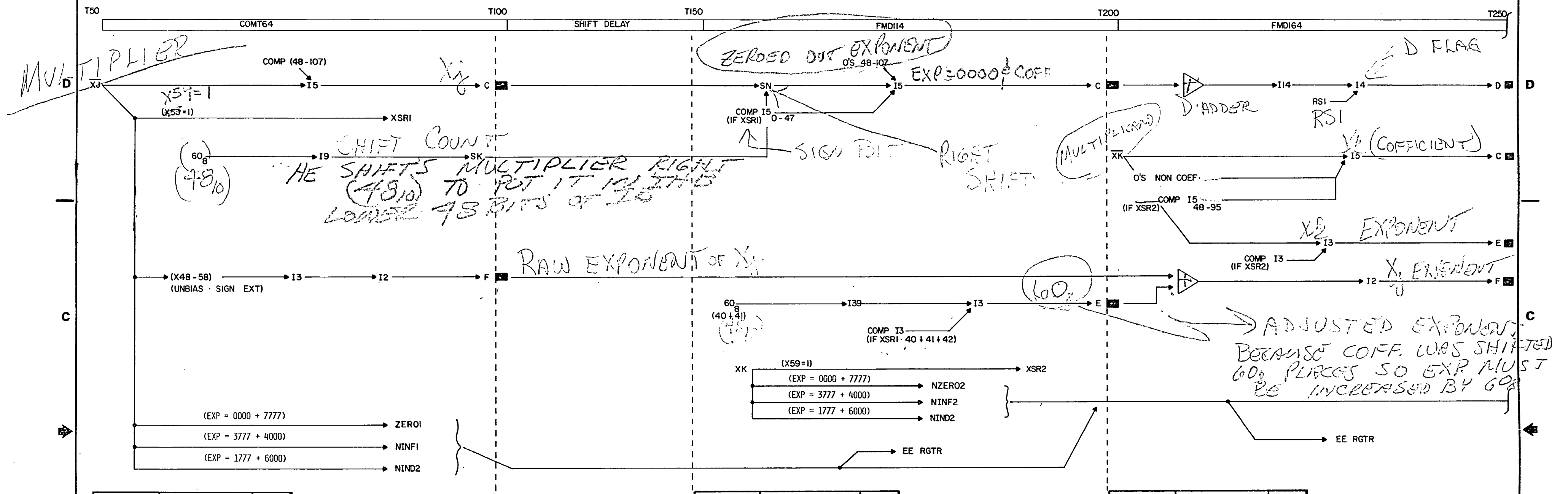
CONTROL DATA
CANADIAN DEVELOPMENT DIVISION

INSTRUCTION FLOW
SEQUENCE: FAD
INSTRUCTION: 30,32,31,33,34,35

CODE IDENT		DWG. NO.	REV
34570	D	19981800	A
FIGURE 5-2-24		PAGE NO. 5-2-50-5	

CARLETON INSTRUMENTS 138-2 300-12-74

SHIFTS & ADDS



TERM NAME	COMMAND OR FUNCTION	DPD NO
CØM50		15
SELXJ	SELECT XJ RGTR	13
CØM64		15
I5C85	COMP I5 48-95	8
I5C67	COMP I5 96-107	8
CØMT64		15
$\overline{SLI91}\cdot\overline{SLI90}$	$60_8 \longrightarrow$ I9	9
NCØ64		15
SK1·SK2	PRESET SK CNTR	9
CØM50		15
ENABLC	ENABLE C RGTR	8
CØM64		15
XSR1	(IF X59=1)	24
COMT64	X48-58 (EXP)	15
CEXPI3	UNBIAS SIGN EXT	15
SEXPI3	$\longrightarrow$ I3 0-17	12
NCØ64		
I3I2	I3 $\longrightarrow$ I2	13
ENABF	ENABLE F RGTR	13
CØM64		
NINFI	SET NINFI (IFEXP=3777+4000)	24
NINDI	SET NINDI (IF EXP= 1777+6000)	24
NZERO1	SET NZERO1 (IF EXP= 0000+7000)	24
CØM64		
BØN	SET BØN FF (IF X59 ≠ 47)	24
NCØ64		
I5185	$X_{0-59} \longrightarrow I5_{48-107}$	7
I5285		7
I5164		7
I5264	$(I5S4 \cdot I5S2 \cdot \overline{I5S1})$	7
I5236		7
I5436		7

TERM NAME	COMMAND OR FUNCTION	DPD NO
$\overline{I5S1}\cdot\overline{I5S2}$ $\overline{I5S4}$	O'S $\longrightarrow$ NON COEF (I5 48-107)	8
FM114·XSR1		
FI5C06	COMP I5 0-46	25
FI5C97	COMP I5 47	25
FM114		
FENBC	ENABLE C RGTR	25
FMD100		24
FMSXK	SELECT XK RGTR	24
FM114		24
ENXSR2	SET XSR2	24
XSR2	(IF X59·1)	24
NINF2	SET NINF2 (IF EXP= 3777+4000)	24
NIND2	SET NIND2 (IF EXP= 1777+6000)	24
NZERO2	SET NZERO2 (IF EXP= 0000+7777)	24
FM114	CLR BØN FF	
BØN	(IF BON·X59= X47)	24
NIMP	$\left\{ \left[40\text{-}42 \cdot \overline{BON} \cdot \overline{NZERO1} \cdot \overline{NZERO2} \right]\right.$ BLOCK END CASE EXIT FOR INTEGER MULTIPLY	25
FM114		
CONST4	[40+41]	
CONST5	$60_8 \longrightarrow$ I39	1
FM114		
FI39I3	I39 $\longrightarrow$ I3	26
SI39I3		12
FM114	[XSR1·40 + 41 + 42]	24
FAMI3C		26
SCØMI3	COMP I3	12
FM114		
FENBE	ENABLE E RGTR	26
FM114	[40 + 41 + 42]	
FI5004	$SN_{0-47} \longrightarrow I5_{0-47}$	25
FI5204		25
FI5057	$(I5S4 \cdot \overline{I5S2} \cdot I5S1)$	25
FI5257		25
FM114		
FENBEE	ENABLE EE RGTR	26

TERM NAME	COMMAND OR FUNCTION	DPD NO
$\overline{FDI14}$	D ADD $\longrightarrow$ I14	26
FM164		
I4-20·I4-21	I14 $\longrightarrow$ I4, RS1 $\longrightarrow$ I4	26
FMD150		24
FMSXK	SELECT XK RGTR	24
$\overline{I5S4}\cdot\overline{I5S2}$ $\cdot\overline{I5S1}$	O'S NON COEF (I5 0-47, 96-107)	
FM164	$[\overline{XSR2}]$	24
FI5C85	COMP I5 48-95	25
FM164	X48-58 (EXP)	
FEXPI3	UNBIAS·SIGN EXT	26
SEXPI3	$\longrightarrow$ I3 0-17	12
FM164	[XSR2·40 + 41 + 42]	
FAMI3C		26
SCØMI3	COMP I3	12
FM164		
FENBF	ENABLE F RGTR	26
FENBE	ENABLE E RGTR	26
FM164		
FENBD	ENABLE D RGTR	26
FENBC	ENABLE C RGTR	26
FM164		
FI5185	$X_{0-47} \longrightarrow I5_{48-95}$	
FI5285	$(I5S4\ I5S2\ \overline{I5S1})$	
FM164	40 + 40	
FI5C67	COMP I5 96-107	
FM164		
FENBEE	ENABLE EE RGTR	26

(Part 1 of 2)

CANADIAN DEVELOPMENT DIVISION

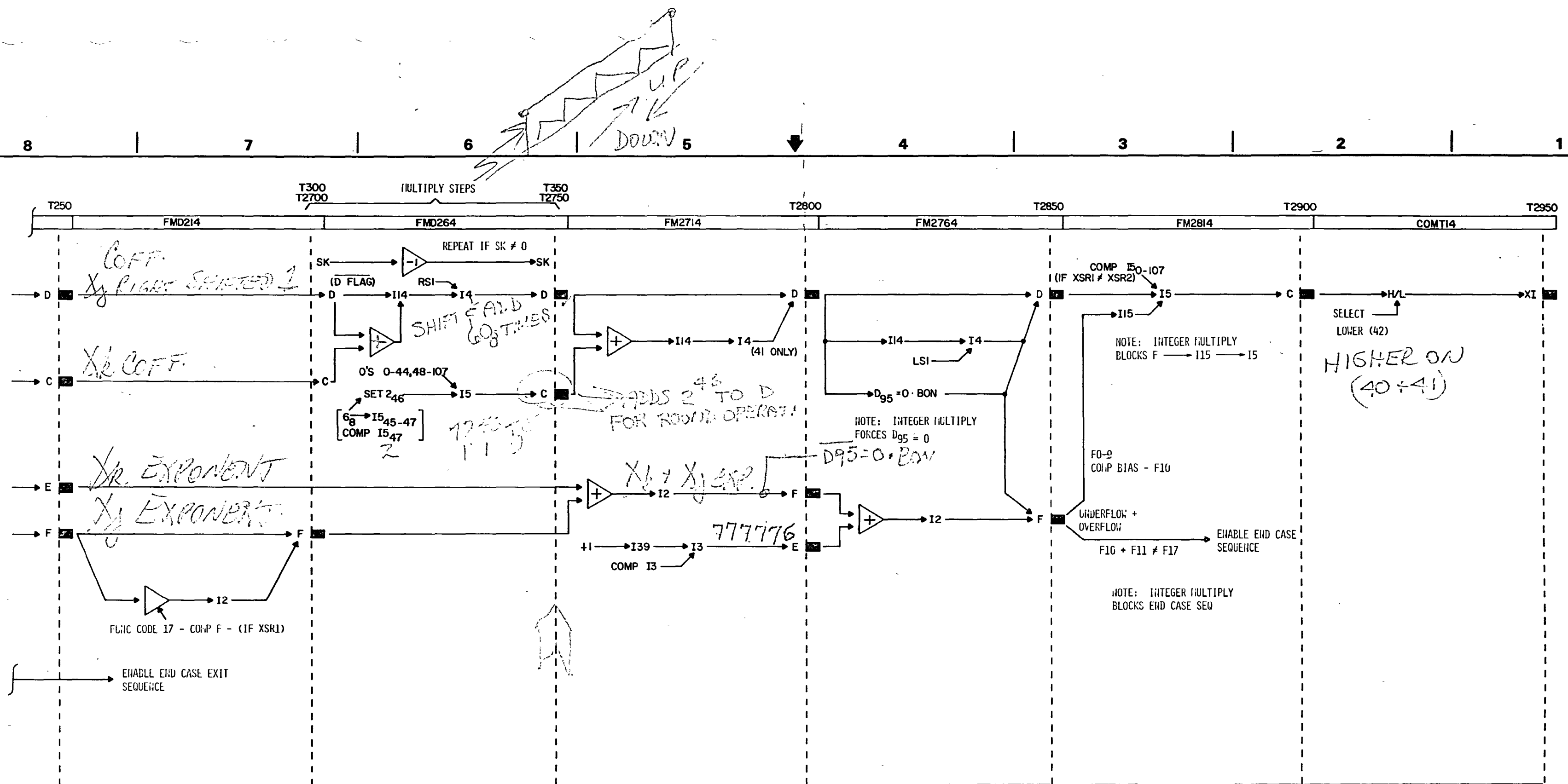
INSTRUCTION FLOW SEQUENCE : FMD INSTRUCTION : 40, 41, 42

CODE IDENT		DWG. NO.	REV
34570	D	19981800	A

FIGURE 5-2-25

PAGE NO. 5-2-50·6

CARLETON INSTRUMENTS 138-3 200-12 74



TERM NAME	COMMAND OR FUNCTION	DPD NO
FM214	[XSR1]	
FENBF	ENABLE F RGTR	26

TERM NAME	COMMAND OR FUNCTION	DPD NO
$\overline{FM264}$		
SKS1·SKS2	DECREMENT SK COUNTER	9
FM264	[$\overline{D\ FLAG}$· 40+41+42]	
FDI14	D RGTR → I14	26
$\overline{FDI14}$	D ADD → I14	26
FM264	I14 → I4	26
I4-20·I4-21	RS1 → I4	
FM264	SKEQ0	
FENBC	ENABLE C RGTR	25
$\overline{I5S4}$·$\overline{I5S2}$ ·I5S1	0'S → $I5_{0-44}$ 48-107	8
FM264		
FI5057	$6_8 \rightarrow I5_{45-47}$	25
FI5157		25
FI5257	COMP $I5_{47}$	25
FI5C47		25
FM264		
FENBD	ENABLE D RGTR	26

TERM NAME	COMMAND OR FUNCTION	DPD NO
$\overline{FDI14}$	D ADDER → I14	26
F2714	41	
FENBD	ENABLE D RGTR	26
F2714		
CONSTO	+1 → I39	1
F2714		
FI3913	I39 → I3	26
SI3913		12
F2714		
FAMI3C	COMP I3	26
SCØMI3		12
F2714		
FENBF	ENABLE F RGTR	26
FENBE	ENABLE E RGTR	26

TERM NAME	COMMAND OR FUNCTION	DPD NO
F2764		
FDI14	D RGTR → I14	26
F2764		
I4-20·$\overline{I4-21}$	I14 → I4	26
	LS1 → I4	
F2764	[BON·$\overline{D_{95}}$·40+41+42]	
FENBD	ENABLE D RGTR	26
FENBF	ENABLE F RGTR	26
NOTE:	INTEGER MULTIPLY FORCES D95= 0	

TERM NAME	COMMAND OR FUNCTION	DPD NO
F2814		
FI5004	$D_{0-95} \rightarrow I5_{0-95}$	25
FI5057		25
FI5085	($\overline{I5S4}$·$\overline{I5S2}$·I5S1)	25
FI5267	$F_{0-9} \xrightarrow{\overline{F10}} I15_{48-59}$	25
FEXI5		8
FI5267	$I15_{48-59} \rightarrow I5_{96-107}$	25
F2814		
FENBC	ENABLE C RGTR	25
F2814		
FMDEXT	ENABLE RNI COMMON TIME	24
F2814		
FØNF7B	F10+F11+F17	
FMENDC	ENABLE END CASE EXIT	24
NOTES:	INTEGER MULTIPLY BLOCKS F I15 I5 AND END CASE EXIT	
F2814		
FI5C06	[40+41+42	25
FI5C47	XSR1 ≠ XSR2]	25
FI5C85		25
FI5C67	COMP I5 0-107	25

TERM NAME	COMMAND OR FUNCTION	DPD NO
COMX00		15
SELXI	SELECT XI RGTR	2
COMENX	ENABLE WRITE STROBE X RGTR	2
COMX00·42		15
HLSL	SELECT LOWER	10

(Part 2 of 2)

CONTROL DATA
CANADIAN DEVELOPMENT DIVISION

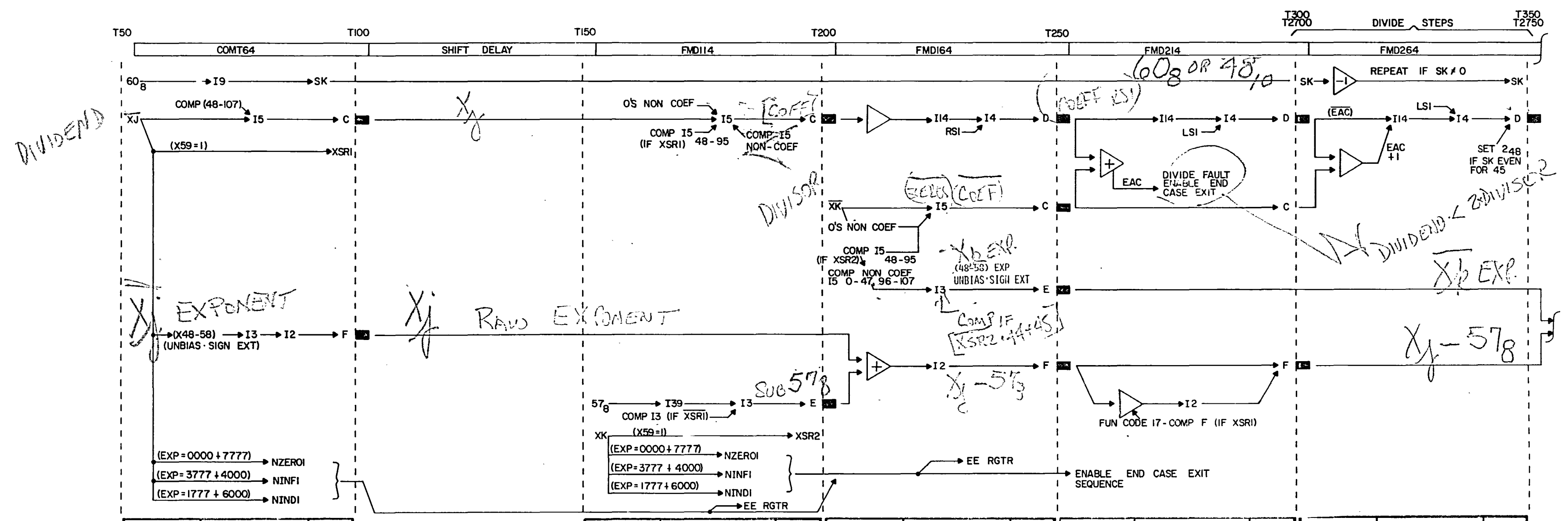
INSTRUCTION FLOW
SEQUENCE : FMD
INSTRUCTION : 40,41,42

CODE IDENT.		DWG. NO.	REV
34570	D	19981800	A
FIGURE 5-2-25		PAGE NO. 5-2-50-7	

CARLETON INSTRUMENTS 138-3 300-1274

SHIFTS & SUBTRACTS

IF EAC SEND BIT TO QUOTIENT



TERM NAME	COMMAND OR FUNCTION	DPD NO
CØM50		15
SELXJ	SELECT XJ RGTR	13
CØM64		15
I5C85	COMP I5 48-95	8
I5C67	COMP I5 96-107	8
COMT64		15
$\overline{SLI91}\cdot\overline{SLI90}$	$60_8 \rightarrow$ I9	9
NCØ64		15
SK1·SK2	ORESET SK CNTR	9
COM50		15
ENABLC	ENABLE C RGTR	8
COM64		15
XSR1	SET XSR1 (IF X59=1)	24
COM64	X48-58 (EXP)	15
CEXPI3	UNBIAS·SIGN EXT	15
SEXPI3	$\rightarrow$ I3 0-17	12
NCØ64		
I3I2	I3 $\rightarrow$ I2	13
ENABF	ENABLE F RGTR	13
COM64		
NINF1	SET NINF1 (IF EXP=3777+4000)	24
NIND1	SET NIND1 (IF EXP=1777+6000)	24
NZERO1	SET NZERO1 (IF EXP=0000+7000)	24
COM64	SET BØN FF (IF X59≠ 47)	24
NCO64		
I5185		7
I5285	$X_{0-59} \rightarrow I5_{48-107}$	7
I5164		7
I5264	(I5S4·I5S2·$\overline{I5S1}$)	7
I5236		7
I5436		7

TERM NAME	COMMAND OR FUNCTION	DPD NO
$\overline{I5S4}\cdot\overline{I5S2}$ $\cdot\overline{I5S1}$	O'S NON COEF (I5 0-47, 96-107)	8
FM114·XSR1		
FI5C85	COMP I5 48-95	25
FM114		
FENBC	ENABLE C RGTR	25
FMD100		24
FMSXK	SELECT XK RGTR	24
FM114		24
ENXSR2	SET XSR2	24
XSR2	(IF X59= 1)	24
NINF2	SET NINF2 (IF EXP= 3777+4000)	24
NIND2	SET NIND2 (IF EXP= 1777+6000)	24
NZERO2	SET NZERO2 (IF EXP= 0000+7777)	24
FM114		
BØN	CLR BØN FF (IF BON= X59= X47)	24
FM114	44 + 45	
CONST0		
CONST1		
CONST2	$57_8 \rightarrow$ I39	1
CONST3		
CONST5		
FM114		
FI39I3	I39 $\rightarrow$ I3	26
SI39I3		12
FM114		
FENBE	ENABLE E RGTR	26
FM114		
FENBEE	ENABLE EE RGTR	26

TERM NAME	COMMAND OR FUNCTION	DPD NO
$\overline{FDI14}$	D ADD $\rightarrow$ I14	26
FM164		
$\overline{I4\text{-}20}\cdot$I4-21	I14 $\rightarrow$ I14, RS1	26
FMD150		24
FMSXK	SELECT XK RGTR	24
$\overline{I5S4}\cdot\overline{I5S2}$ $\cdot\overline{I5S1}$	O'S $\rightarrow$ NON COEF (I5 0-47, 96-107)	8
FM164	X48-58 (EXP)	
FEXPI3	UNBIAS·SIGN EXT	26
SEXPI3	$\rightarrow$ I3 0-17	12
FM164		
FAMI3C	[$\overline{XSR2}$·44+45]	26
SCØMI3	COMP I3	12
FM164		
FENBF	ENABLE F RGTR	26
FENBE	ENABLE E RGTR	26
FM164		
FENBD	ENABLE D RGTR	26
FENBC	ENABLE C RGTR	26
FM164		
FI5185	$X_{0-47} \rightarrow I5_{48-95}$	25
FI5285	(I5S4·I5S2·$\overline{I5S1}$)	25
FM164	[44 + 45]	
FI5C06		25
FI5C47	COMP I5 0-47,	25
FI5C67	96-107	25
FM164		
FENBEE	ENABLE EE RGTR	26
FM164	[XSR2]	24
FI5C85	COMP I5 48-95	25

TERM NAME	COMMAND OR FUNCTION	DPD NO
FM214	[44+45·$\overline{CRY107}$]	
FDI14	D RGTR $\rightarrow$ I14	26
NILEAC		
SET FEC	DIVIDE FAULT ENABLE END CASE PLOCK FMD264	24
FM214		
I4-20·$\overline{I4\text{-}21}$	I14 $\rightarrow$ I4, LS1	26
FM214	[XSR1]	
FENDF	ENABLE F RGTR	26
FM214		
FENDD	ENABLE D RGTR	26

TERM NAME	COMMAND OR FUNCTION	DPD NO
FM264		9
SKS1·$\overline{SKS2}$	DECREMENT SK COUNTER	
FM264		
FDI14	[44 + 45·$\overline{CRY107}$]	26
I14S	D RGTR $\rightarrow$ I14	5
$\overline{I14S}$	[44 + 45·CRY107]	5
	D ADD $\rightarrow$ I14	
FM264		
I4-20·$\overline{I4\text{-}21}$	I14 $\rightarrow$ I4, LS1	26
FM264		
FENBD	ENABLE D RGTR	26
FM264		
1TØ40	[$\overline{SK00}$·45] 1 $\rightarrow$ D48	5

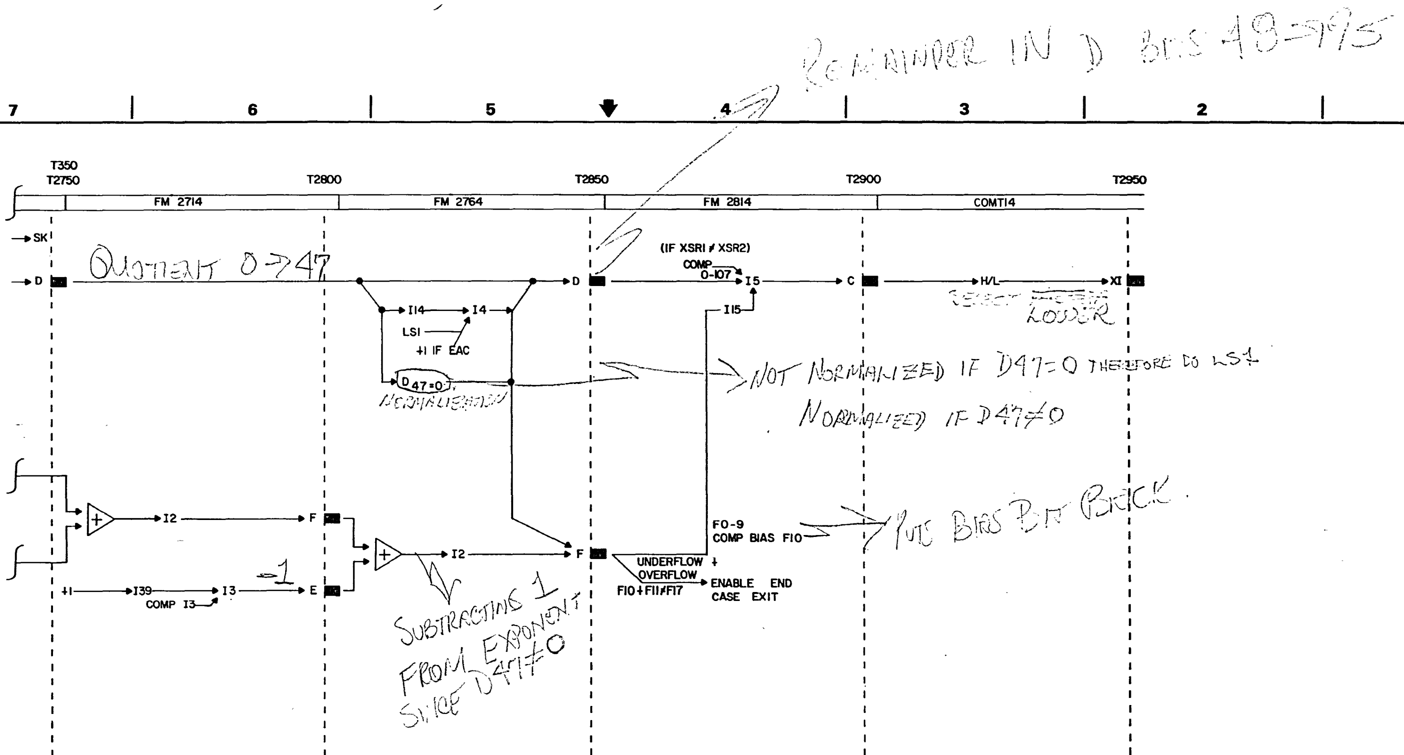
(Part 1 of 2)

CONTROL DATA
CANADIAN DEVELOPMENT DIVISION

INSTRUCTION FLOW
SEQUENCE: FMD
INSTRUCTION: 44, 45

CODE IDENT		DWG NO	REV
34570	D	19981800	A

FIGURE 5-2-26



TERM NAME	COMMAND OR FUNCTION	DPD NO
F2714		
CONSTO	$+1 \longrightarrow I39$	1
F2714		
FI39I3	$I39 \longrightarrow I3$	26
SI39I3		12
F2714		
FAMI3C	COMP I3	26
SCØMI3		12
F2714		
FENBF	ENABLE F RGTR	26
FENBE	ENABLE E RGTR	26

TERM NAME	COMMAND OR FUNCTION	DPD NO
F2764		
FDI14	D RGTR $\longrightarrow$ I14	26
F2764		
$I4\text{-}20 \cdot \overline{I4\text{-}21}$	$I14 \longrightarrow I4$, LS1	26
F2764	$[\overline{D_{47}} \cdot 44+45]$	
FENBD	ENABLE D RGTR	26
FENBF	ENABLE F RGTR	26

TERM NAME	COMMAND OR FUNCTION	DPD NO
F2814		
FI5004	$D_{0-95} \longrightarrow I5_{0-95}$	25
FI5057		25
FI5085	$(\overline{I5S4} \cdot \overline{I5S2} \cdot I5S1)$	25
FI5267	$F_{0-9} \longrightarrow I15_{48-59}$	25
FEXI5	$\overline{F}_{10}$	8
FI5267	$I15_{48-59} \longrightarrow I5_{96-107}$	25
F2814		
FI5C06	[44 + 45. XSR1≠ XSR2]	25
FI5C47		25
FI5C85		25
FI5C67	COMP $I5_{0-107}$	25
F2814		
FENBC	ENABLE C RGRT	25
F2814		
FMDEXT	ENABLE RNI, COMMON TIME	24
F2814		
FØNF7B	F10 + F11 = F17	
FMENDC	ENABLE END CASF FXIT	24

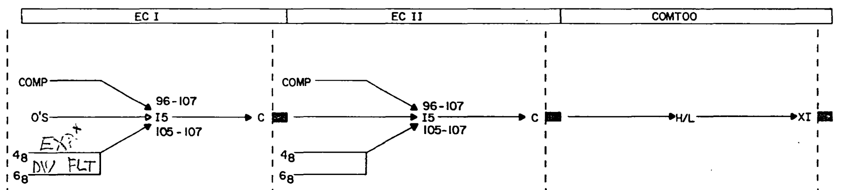
TERM NAME	COMMAND OR FUNCTION	DPD NO
COMX00		15
SELXI	SELECT XI RGTR	2
COMENX	ENABLE WRITE STROBE X RGTR	2
~~HLSL~~	~~SELECT HIGHER~~	10
HLSL	SELECT LOWER	

(Part 2 of 2)

CONTROL DATA
CANADIAN DEVELOPMENT DIVISION

INSTRUCTION FLOW
SEQUENCE: FMD
INSTRUCTION: 44, 45

CODE IDENT.		DWG. NO.	REV
34570	D	19981800	A
FIGURE 5 - 2 - 26		PAGE NO. 5-2-50-9	



FOR 40 - 42, 44 - 45

$4_8 \longrightarrow I5^{105-107}$ = EXP$^+$

$6_8 \longrightarrow I5^{105-107}$ = DIVIDE FLT

COMP $I5^{96-107}$ = DIVIDE FLT + (LIKE SIGNS·EXP$^+$)

FOR 30 - 35

$4_8 \longrightarrow I5^{105-107}$ = INF OP

$6_8 \longrightarrow I5^{105-107}$ = IND OP +

[BOTH OPS INF·(SUB·LIKE SIGNS) + (ADD·UNLIKE SIGNS)]

COMP $I5^{96-107}$ = ($6_8 \longrightarrow$ I5) + (XJ = $\pm\infty$) + (ADD·XK = $\mp\infty$) + (SUB·XK = $-\infty$)

FOR 40 - 42, 44 + 45

$4_8 \longrightarrow I5^{105-107}$ = MULT·(XJ + XK = ∞) + DIVIDE·(XJ + XK = 0)

$6_8 \longrightarrow I5^{105-107}$ = IND OP + [MULT·(XJ ∞·XKO) + (XJO·XK∞)] + [DIVIDE·(XJ·XK = 0) + (XJ·XK = ∞)]

COMP $I5^{96-107}$ = ($6_8 \longrightarrow$ I5) + [LIKE SIGNS·(XJ + XK = ∞)]

END CASE RESULTS

RESULTS

0000	0——0	= ZERO
3777	0——0	= ~~+00~~ +∞
4000	0——0	= ~~-00~~ -∞
1777	0——0	= IND

5-12 OF REFERENCE

TERM NAME	COMMAND OR FUNCTION	DPD NO
FAENDC	[IND + INF + UNDERFLOW]	24
FMENDC	[IND + INF + UNDERFLOW + ZERO OPERAND + DIVIDE FAULT]	24
ENDC	ENABLE END CASE EXIT (ECI)	25
I5S4·I5S2 ·I5S1	O'S $\longrightarrow$ I5	8
ECI		
EI510	$4_8 \longrightarrow I5_{105-107}$	25
EI5257	$6_8 \longrightarrow I5_{105-107}$	25
ECI567	COMP $I5_{96-107}$	25
ECI		
ECENBC	ENABLE C RGTR	25

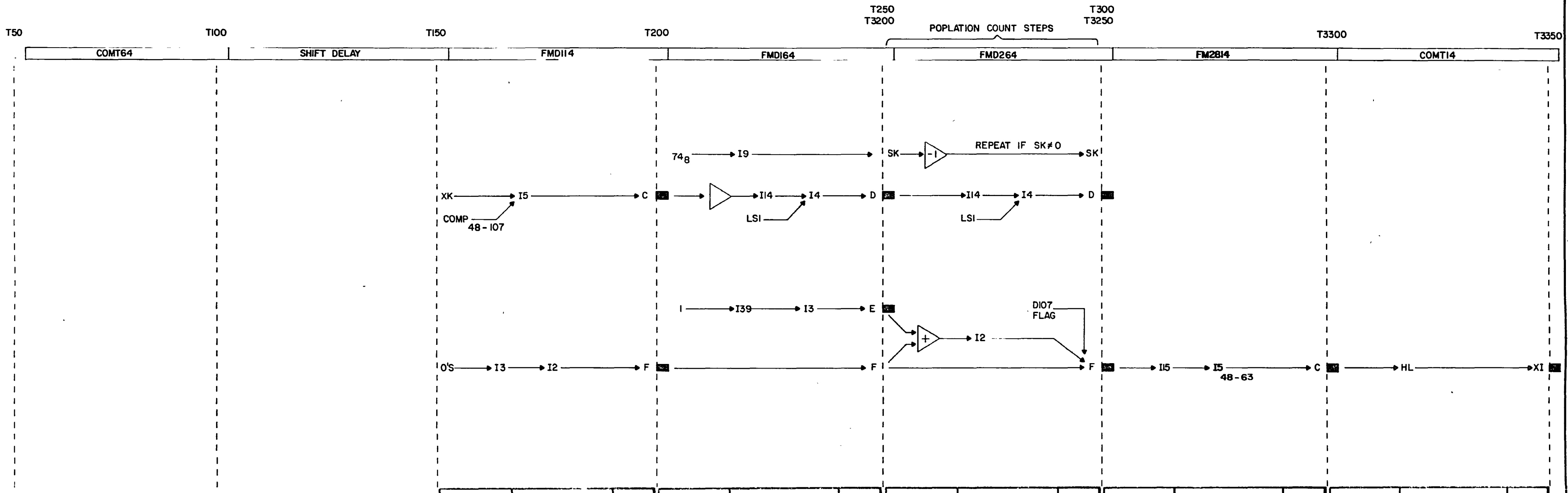
TERM NAME	COMMAND OR FUNCTION	DPD NO
ECII		
EI510	$4_8 \longrightarrow I5_{105-107}$	25
EI5257	$6_8 \longrightarrow I5_{105-107}$	25
ECI567	COMP $I5_{96-107}$	25
ECII	IND + INF + ZERO	
ECENBC	ENABLE C RGTR	25
ECII		
ECEXIT	ENABLE RNI, COMMON TIME	25

TERM NAME	COMMAND OR FUNCTION	DPD NO
CONXOO		15
SELXI	SELECT XI RGTR	2
COMENX	ENABLE WRITE STROBE X RGTR	2

CONTROL DATA CANADIAN DEVELOPMENT DIVISION

END CASE EXIT SEQ

CODE IDENT.		DWG. NO.	REV
34570	D	19981800	A
FIGURE 5-2-27		PAGE NO. 5-2-50-10	



TERM NAME	COMMAND OR FUNCTION	DPD NO
FM114	[FUN47]	
FI5185		25
FI5285	$X_{0-59} \rightarrow I5_{48-107}$	25
FI5167		25
FI5267	$(I5S4 \cdot I5S2 \cdot \overline{I5S1})$	25
FM114	[FUN47]	
FM4715	COMP I5 48-107	24
FI5C85		25
FM114		
FENBC	ENABLE C RGTR	25
DEFAULT	0'S → I3	12
FM114	FUN47	
FI3I2	I3 → I2	26
FNABF	ENABLE F RGTR	13

TERM NAME	COMMAND OR FUNCTION	DPD NO
$\overline{SLI90} \cdot \overline{SLI91} \cdot \overline{COM64}$	$74_8 \rightarrow$ I9	9
$\overline{FDI14}$	D ADD → I14	26
FM164		
$I4\text{-}20 \cdot \overline{I4\text{-}21}$	I14 → I4; LS1 → I4	26
FM164		
FENBD	ENABLE D RGTR	26
FM164	[FUN47]	
CONSTO	+1 → I39	1
FM164	[NFUN47]	
FI39I3	I39 → I3	26
FM164		
FENBE	ENABLE E RGTR	26

TERM NAME	COMMAND OR FUNCTION	DPD NO
FM264		
$SKS1 \cdot \overline{SKS2}$	DECREMENT SK COUNTER	9
$\overline{FDI14}$	D ADD → I14	26
FM264	[47]	
I4-20·I4-21	I14 → I4; LS1 → I4	26
FM264		
FENBD	ENABLE D RGTR	26
FM264	[47·D FLAG]	
FENBF	ENABLE F RGTR	26

TERM NAME	COMMAND OR FUNCTION	DPD NO
$\overline{F46X}$·	$F_{0-17} \rightarrow I15_{0-17}$	8
F2814		
FI5285	$I15 \rightarrow I5_{48-63}$	25
F2814		
FENBC	ENABLE C RGTR	25
F2814		
FMDEXT	ENABLE RNI, COMMON TIME	24

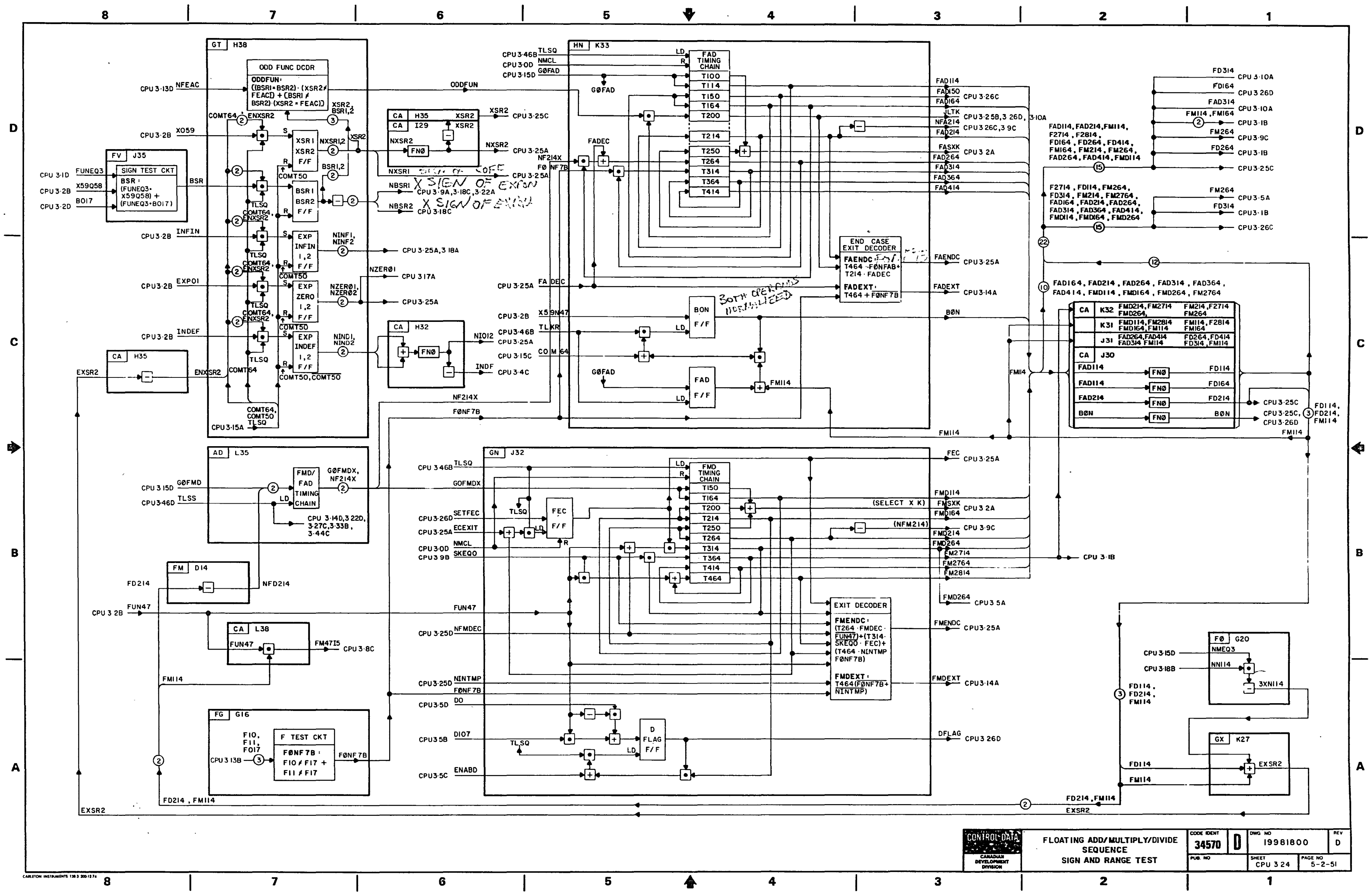
TERM NAME	COMMAND OR FUNCTION	DPD NO
COMX00		15
SELXI	SELECT XI RGTR	2
COMENX	ENABLE WRITE STROBE X RGTR	2

CONTROL DATA
CANADIAN DEVELOPMENT DIVISION

INSTRUCTION FLOW
SEQUENCE: FMD
INSTRUCTION: 47

CODE IDENT.		DWG NO	REV
34570	D	19981800	A
FIGURE 5-2-28		PAGE NO 5-2-50-11	

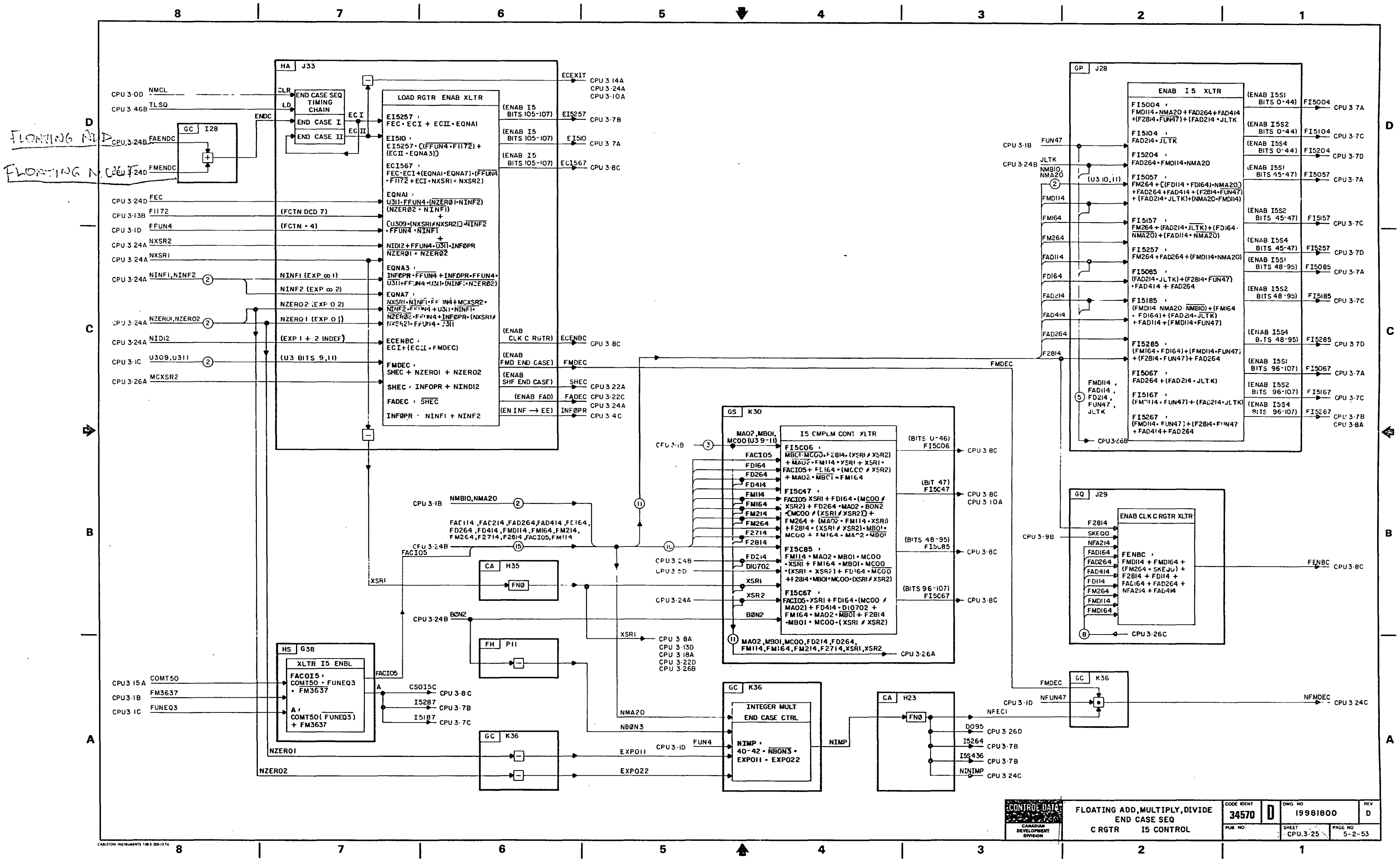
CARLETON INSTRUMENTS 138-3 300-12-74



CONTROL DATA
CANADIAN DEVELOPMENT DIVISION

FLOATING ADD/MULTIPLY/DIVIDE SEQUENCE
SIGN AND RANGE TEST

CODE IDENT		DWG NO	REV
34570	D	19981800	D
PUB NO		SHEET CPU 3 24	PAGE NO 5-2-51



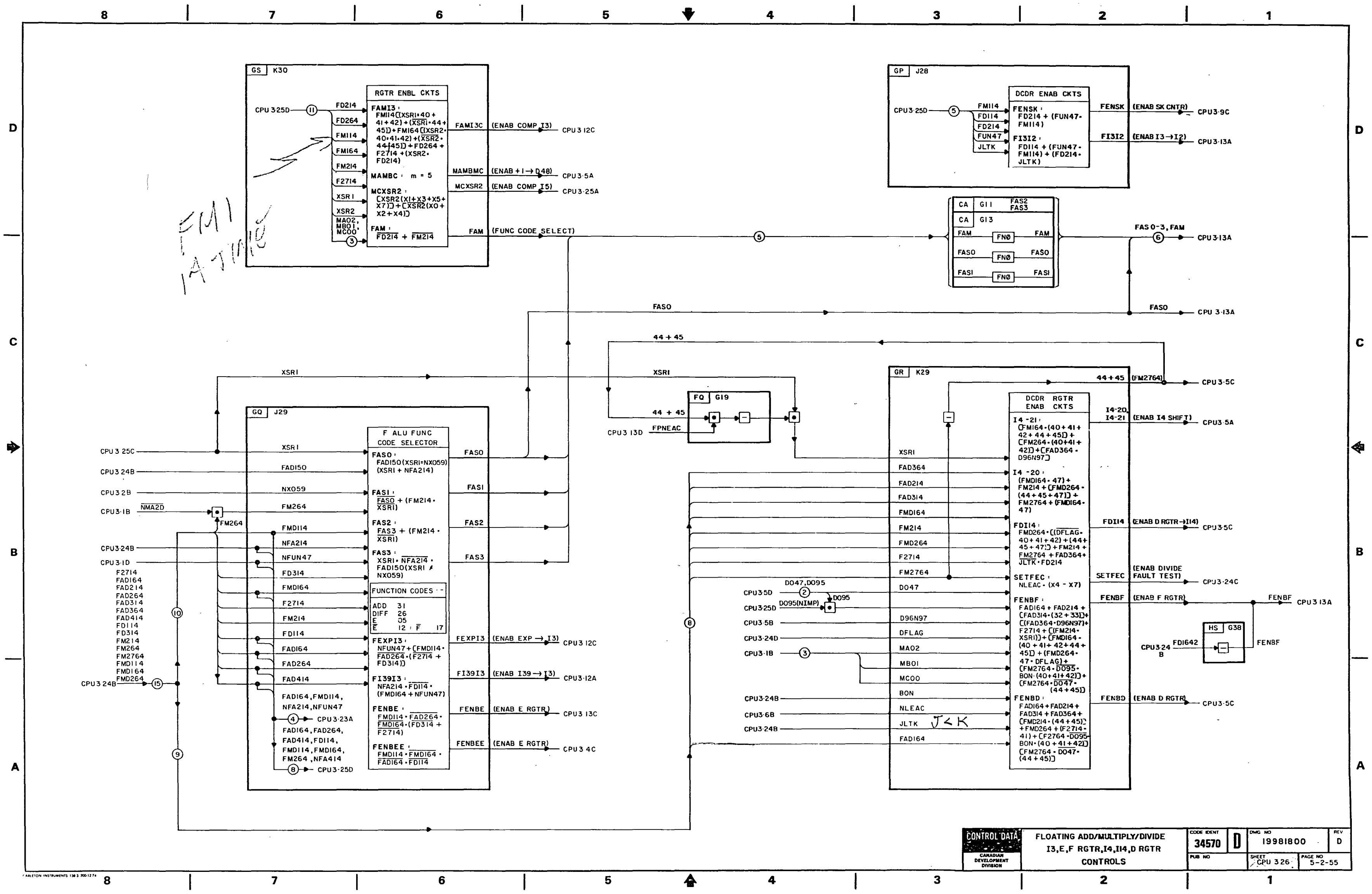
FLOATING ADD
FLOATING MULT
HA J33
END CASE SEQ TIMING CHAIN
END CASE I
END CASE II
LOAD RGTR ENAB XLTR
EI5257 = FEC·ECI + ECII·EQNAI
EI510 = EI5257·[(FFUN4·FI172) + (ECII·EQNA3)]
ECI567 = FEC·ECI+(EQNAI·EQNA7)·(FFUN4·FI172+ECI·NXSRI=NXSR2)
ECENBC = ECII+(ECII·FMDEC)
FMDEC = SHEC + NZEROI + NZERO2
SHEC = INFOPR + NINDI2
FADEC = SHEC
INFOPR = NINFI + NINF2
GC I28
GP J28
ENAB I5 XLTR
GS K30
I5 CMPLM CONT XLTR
GQ J29
ENAB CLK C RGTR XLTR
FENBC = FMDI14 + FMDI64 + (FM264 + SKEQO) + F2814 + FDI14 + FACI64 + FAD264 + NFA214 + FAD414
HS G38
XLTR I5 ENBL
FACOI5 = COMT50 · FUNEQ3 · FM3637
A = COMT50(FUNEQ3) + FM3637
CA H35
FH P11
GC K36
INTEGER MULT END CASE CTRL
NIMP = 40-42·NBON3·EXPO11·EXPO22
CA H23
GC K36

CONTROL DATA CANADIAN DEVELOPMENT DIVISION

FLOATING ADD, MULTIPLY, DIVIDE END CASE SEQ
C RGTR I5 CONTROL

CODE IDENT		DWG NO	REV
34570	D	19981800	D
PUB NO		SHEET CPU.3·25	PAGE NO 5-2-53

CARLETON INSTRUMENTS 138-3 300-1274



GS K30
RGTR ENBL CKTS
FAMI3 : FMI14(XSR1·40+41+42)+(XSR1·44+45)+FMI64(XSR2·40·41·42)+(XSR2·44+45)+FD264+F2714+(XSR2·FD214)
MAMBC : m = 5
MCXSR2 : (XSR2(X1+X3+X5+X7))+(XSR2(X0+X2+X4))
FAM : FD214 + FM214
FAMI3C (ENAB COMP I3) CPU3 12C
MAMBMC (ENAB +1→D48) CPU3 5A
MCXSR2 (ENAB COMP I5) CPU3 25A
FAM (FUNC CODE SELECT)
CPU3 25D
FD214
FD264
FMI14
FMI64
FM214
F2714
XSR1
XSR2
MA02, MB01, MC00
FMI
IA TIME
GP J28
DCDR ENAB CKTS
FENSK : FD214 + (FUN47·FMI14)
FI3I2 : FDI14 + (FUN47·FMI14) + (FD214·JLTK)
FENSK (ENAB SK CNTR) CPU3 9C
FI3I2 (ENAB I3→I2) CPU3 13A
FMI14
FDI14
FD214
FUN47
JLTK
CA G11 FAS2 FAS3
CA G13
FAM FN0 FAM
FAS0 FN0 FAS0
FAS1 FN0 FAS1
FAS 0-3, FAM CPU3 13A
FAS0
FAS0 CPU 3 13A
44 + 45
XSR1
FQ G19
44 + 45
CPU3 13D FPNEAC
GQ J29
F ALU FUNC CODE SELECTOR
FAS0 : FADI50(XSR1+NX059)(XSR1 + NFA214)
FAS1 : FAS0 + (FM214·XSR1)
FAS2 : FAS3 + (FM214·XSR1)
FAS3 : XSR1·NFA214·FADI50(XSR1 ≠ NX059)
FUNCTION CODES
ADD 31
DIFF 26
E 05
E 12 : F 17
FEXPI3 : NFUN47+(FMDI14·FAD264·(F2714 + FD314))
FI39I3 : NFA214·FDI14·(FMDI64 + NFUN47)
FENBE : FMDI14·FAD264·FMDI64·(FD314 + F2714)
FENBEE : FMDI14·FMDI64·FADI64·FDI14
FAS0
FAS1
FAS2
FAS3
FEXPI3 (ENAB EXP → I3) CPU3 12C
FI39I3 (ENAB I39 → I3) CPU3 12A
FENBE (ENAB E RGTR) CPU3 13C
FENBEE (ENAB E RGTR) CPU3 4C
CPU3 25C XSR1
CPU3 24B FADI50
CPU3 2B NX059
CPU3 1B NMA2D FM264
FM264
FMDI14
CPU3 24B NFA214
CPU3 1D NFUN47
FD314
FMDI64
F2714
FM214
FDI14
FADI64
FAD264
FAD414
F2714 FADI64 FAD214 FAD264 FAD314 FAD364 FAD414 FDI14 FD314 FM214 FM264 FM2764 FMDI14 FMDI64 FMD264
CPU3 24B
FADI64, FMDI14, NFA214, NFUN47
CPU3 23A
FADI64, FAD264, FAD414, FDI14, FMDI14, FMDI64, FM264, NFA414
CPU3 25D
GR K29
DCDR RGTR ENAB CKTS
I4-21 : (FMI64·(40+41+42+44+45))+(FM264·(40+41+42))+(FAD364·D96N97)
I4-20 : (FMDI64·47)+FM214+(FMD264·(44+45+47))+FM2764+(FMDI64·47)
FDI14 : FMD264·((DFLAG·40+41+42)+(44+45+47))+FM214+FM2764 + FAD364+JLTK·FD214
SETFEC : NLEAC·(X4 - X7)
FENBF : FADI64 + FAD214 + (FAD314·(32+33))+((FAD364·D96N97)+F2714+((FM214·XSR1))+(FMDI64·(40+41+42+44+45))+(FMD264+(F2714·47·DFLAG))+(FM2764·D095)+BON·(40+41+42))+(FM2764·D047·(44+45))
FENBD : FADI64+FAD214+FAD314+FAD364+(FMC214·(44+45))+FMD264+(F2714·41)+(F2764·D095)+BON·(40+41+42)+(FM2764·D047·(44+45))
44+45 (FM2764) CPU3 5C
I4-20, I4-21 (ENAB I4 SHIFT) CPU3 5A
FDI14 (ENAB D RGTR→I14) CPU3 5C
SETFEC (ENAB DIVIDE FAULT TEST) CPU3 24C
FENBF (ENAB F RGTR) FENBF CPU3 13A
FENBD (ENAB D RGTR) CPU3 5C
XSR1
FAD364
FAD214
FAD314
FMDI64
FM214
FMD264
F2714
FM2764
D047
D96N97
DFLAG
MA02
MB01
MC00
BON
NLEAC
JLTK
FADI64
J<K
CPU3 5D D047, D095
CPU3 25D D095(NIMP) D095
CPU3 5B
CPU3 24D
CPU3 1B
CPU3 24B
CPU3 6B
CPU3 24B
HS G38
CPU3 24B FDI642
FENBF
8 7 6 5 4 3 2 1
D C B A

CONTROL DATA CANADIAN DEVELOPMENT DIVISION	FLOATING ADD/MULTIPLY/DIVIDE I3,E,F RGTR,I4,I14,D RGTR CONTROLS	CODE IDENT 34570	D	DWG NO 19981800	REV D
		PUB NO		SHEET CPU 326	PAGE NO 5-2-55

DETAILED PAK DIAGRAM (CPU 3.27)

ECS SUBSYSTEM SEQUENCE

Detection of an ECS instruction (011jK or 012jK) in parcels 0 and 1 causes an ECS request to be sent to the ECS coupler. The continuation of the ECS sequence is suspended until the accept (ECSACP) is returned. The ECS sequence is responsible for making address range tests for both the ECS address and the CM address. This is done in each case by comparing the last word address against the field length (FLE or FL). A test for negative word count is also performed. Violation of any of these conditions causes an address range error (AOR) and aborts the ECS sequence. An attempt to execute an ECS instruction from the wrong parcel, or when no ECS coupler is present, forces an illegal instruction fault.

The ECS sequence sends the word count (Bj + K) and the ECS starting address (X0 0-23 + RAE) to the ECS coupler, and the starting address (A0 + RA) to CMC. If none of the abort conditions are present, a start transfer is sent to the coupler to initiate movement of data. The CPU remains idle during the transfer.

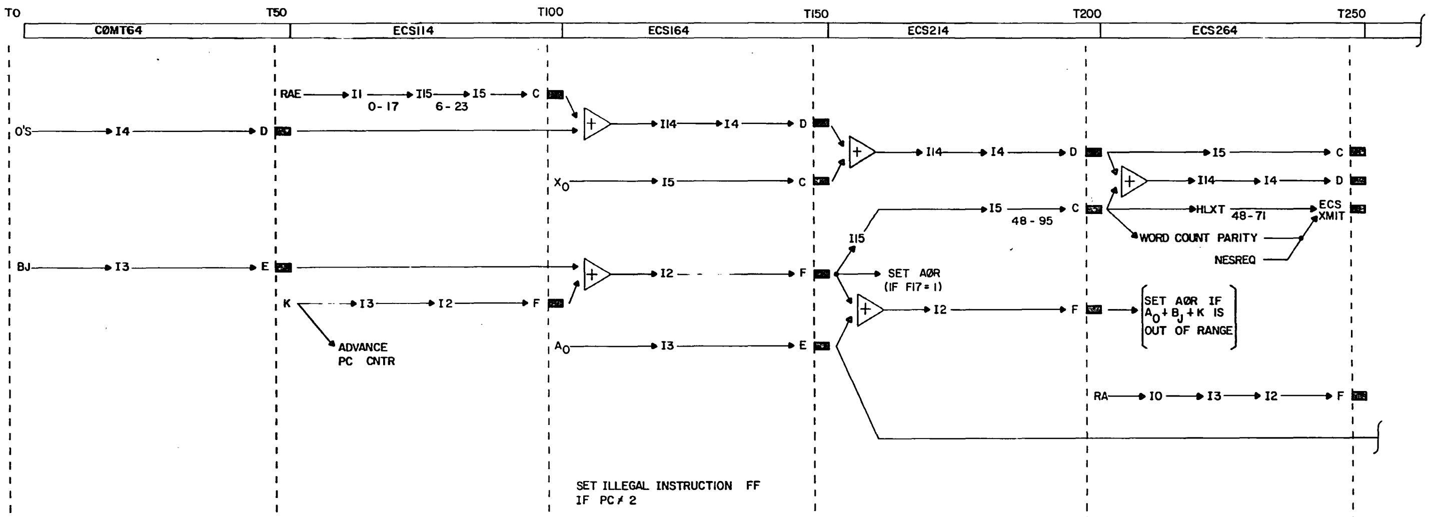
An error exit (ERRABT) or normal exit (ENDTRC) will be sent to the CPU at the completion of the data transfer. The error exit causes the processor to execute the instruction (usually a branch) that is in parcels 2 and 3 of the ECS instruction word. A normal exit bypasses this instruction and does an initial start RNI to the next word.

FLAG REGISTER

The ECS subsystem also contains a flag register primarily used for communication between processors attached to ECS. Access to this register is through an ECS instruction identified by both bit 23 of X0 and bit 23 of FLE being set. The ECS sequence must be modified when a flag operation is detected. The contents of X0 0-23 are sent to the coupler without the addition of RAE. No data transfer is made; however, the coupler will respond with either error or normal exit depending on the flag function bits in X0 and the condition of the flag register.

EXCHANGE BREAKIN

If an exchange request arrives during the execution of an ECS transfer, the ECS instruction is terminated. No provision is made for maintaining addresses or number of words transferred. Consequently, the P register value stored in the exchange package will point to the ECS instruction. It will be reinitiated at the next execution interval of the program as if no ECS data had been processed.



SET ILLEGAL INSTRUCTION FF
IF PC ≠ 2

TERM NAME	COMMAND OR FUNCTION	DPD NO
CØM50		15
SELBJ	SELECT BJ RGTR	2
CØMTC4		15
CØMBI3	$B \rightarrow I3$	12
NCØ50		15
ENABE	ENABLE E RGTR	13
NCØC4		
I40·I41	$O'S \rightarrow I4$	5
NCØ50		
ENABLD	ENABLE D RGTR	5

TERM NAME	COMMAND OR FUNCTION	DPD NO
ECS114		27
SRAECS	$RAE \rightarrow I1$	27
$\overline{F46X}$·ESI15	$I1_{0-17} \rightarrow I15_{6-23}$	8
ECS114		27
ECSI1I5	$I15_{0-59} \rightarrow I5_{48-95}$	27
I5285	$(I5S4 \cdot \overline{I5S2} \cdot \overline{I5S1})$	7
NES114		27
SELKI3	$K \rightarrow I3$	12
ADVPC2	ADVANCE PC CNTR	12
ECS114		27
ESI3I2	$I3 \rightarrow I2$	27
ENABF	ENABLE F RGTR	13

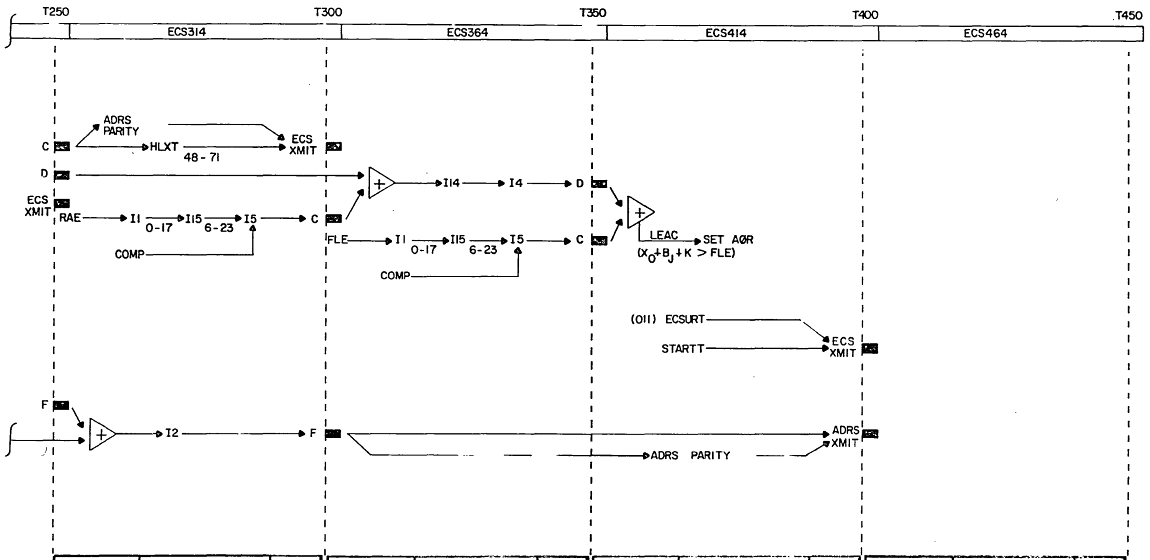
TERM NAME	COMMAND OR FUNCTION	DPD NO
ECS164		
ECSEND	ENABLE D RGTR	27
ECSENF	ENABLE F RGTR	27
NE164		27
I5185	$X_{0-59} \rightarrow I5_{48-107}$	7
ESI1I5		27
I5285	$(I5S4 \cdot I5S2 \cdot \overline{I5S1})$	7
ECS164		
ECSENC	ENABLE C RGTR	27
NE164		27
SELAI3	$A \rightarrow I3$	12
ENABE	ENABLE E RGTR	13

TERM NAME	COMMAND OR FUNCTION	DPD NO
ECS214		
ECSEND	ENABLE D RGTR	27
ECSENF	ENABLE F RGTR	27
$\overline{F46X}$·$\overline{ESI15}$	$F_{0-17} \rightarrow I15_{0-17}$	8
ECS214		27
ESI1I5	$I15_{0-59} \rightarrow I5_{48-95}$	27
I5285	$(I5S4 \cdot \overline{I5S2} \cdot \overline{I5S1})$	7
ECS214		27
ECSENC	ENABLE C RGTR	27
ES214	[F17=1]	27
AØR	SET AØR	17

TERM NAME	COMMAND OR FUNCTION	DPD NO
NE264		27
I5085	$D \rightarrow I5_{48-107}$	7
I5064		7
I5136	$(\overline{I5S4} \cdot \overline{I5S2} \cdot I5S1)$	7
NE264		27
NESREQ	COUPLER REQ →	27
	ECS XMIT	
ECS264		
ECSEND	ENABLE D RGTR	27
EN264		27
ENABØR	ENABLE TEST AØR	17
$\overline{SI00} \cdot \overline{SI01}$	$RA \rightarrow I0$	4
$\cdot \overline{SI02}$		
NE264		27
SLI0I3	$I0 \rightarrow I3$	12
ECS264		27
ESI3I2	$I3 \rightarrow I2$	27
ENABF	ENABLE F RGTR	13

(Part 1 of 2)

CONTROL DATA CANADIAN DEVELOPMENT DIVISION	INSTRUCTION FLOW SEQUENCE : ECS INSTRUCTION : 011, 012	CODE IDENT. 34570	D	DWG. NO. 19981800	REV A
		FIGURE 5-2-29			PAGE NO. 5-2-56-1



TERM NAME	COMMAND OR FUNCTION	DPD NO
ECS314		27
SRAECS	RAE ⟶ I1	27
$\overline{F46X}$·ESI15	$I1_{0-17} \rightarrow I15_{6-23}$	8
ECS314		27
ESI115	$I15_{0-59} \rightarrow I5_{48-95}$	27
I5285		7
ECS314		27
ECSI5C	COMP I5	27
ECS314		
ECSENF	ENABLE F RGTR	27
ECSENC	ENABLE C RGTR	27

TERM NAME	COMMAND OR FUNCTION	DPD NO
ECS364		27
SFLECS	FLE ⟶ I1	27
$\overline{F46X}$·ESI15	$I1_{0-17} \rightarrow I15_{6-23}$	8
ECS364		27
ESI115	$I15_{0-59} \rightarrow I5_{48-95}$	27
I5285		7
ECS364		27
ECSI5C	COMP I5	27
ECS364		
ECSEND	ENABLE D RGTR	27
ECSENC	ENABLE C RGTR	27

TERM NAME	COMMAND OR FUNCTION	DPD NO
ECS414	[NLEAC]	27
SETAØR	SET AØR	17

TERM NAME	COMMAND OR FUNCTION	DPD NO
ECS464	[ENDIRF]	
NESETR	ENABLE RNI	27
	[ERRNBT]	
ESERRX	ENABLE SEQ EXIT	27

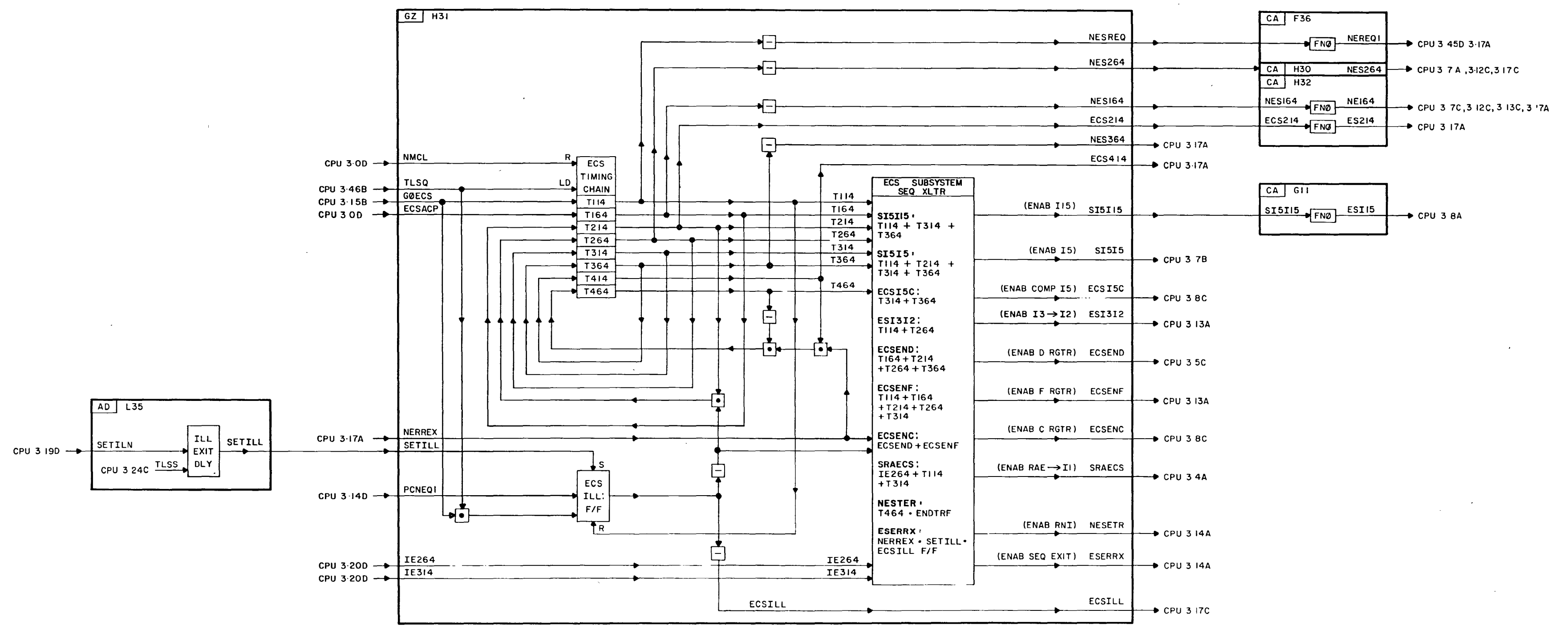
(Part 2 of 2)

CONTROL DATA
CANADIAN DEVELOPMENT DIVISION

INSTRUCTION FLOW
SEQUENCE : ECS
INSTRUCTION : 011, 012

CODE IDENT.		DWG NO	REV
34570	D	19981800	A
FIGURE 5-2-29		PAGE NO 5-2-56.2	

CARLETON INSTRUMENTS 138 3 200-12 74



CONTROL DATA
CANADIAN DEVELOPMENT DIVISION

EXTENDED CORE STORAGE SUBSYSTEM SEQUENCE

CODE IDENT		DWG NO	REV
34570	D	19981800	B
PUB NO		SHEET CPU 3 27	PAGE NO 5-2-57

DETAILED PAK DIAGRAM (CPU3.28)

COMPARE/MOVE DATA SECTION (Part One)

The compare move data section consists of four data registers. Three are shown on diagram 3.28; they are the 54-bit R register, the 60-bit Q register, and the 60-bit S register. The remaining register is shown on diagram 3.29; it is the 48-bit T register.

Q REGISTER

The 60-bit Q register is located on the JC module. It is the primary word formation register. Input to the Q register is through selector I30 which allows selection of either the C register bits 48-107 or R register bits 0-47, 108-113.

R REGISTER

The 54-bit R register is also located on the JC module. It is used as a residue register for data right shifted in the C register prior to storing in the Q register. The input to R comes directly from the C register bits 0-47, 108-113.

S REGISTER

The 60-bit S register is located on the JB module. It acts as a buffer register for data stored in the Q register.

During move instructions (464, 465), data words that have been properly formatted in the Q register are transferred to the S register. The output of S gates directly to the HR register and the output transmitters.

During a compare instruction (466, 467), the S register serves a more useful purpose. Data words that have been properly formatted in the Q register are transferred to the S register awaiting subsequent comparison with corresponding words stored in the Q register.

COMPARISON CIRCUITS

Data comparison is performed on a word basis by the S = Q compare circuit located on the JB module. Each JB module is capable of one character comparison.

The output of the S = Q compare circuit generates a compare character equal signal for each character (COME 0-9). Compare character equal will be at one level when the respective characters in S and Q are equal. The compare character equal signals can also be forced to indicate equality by the force equivalence circuits. These circuits are used by the compare collate instruction exclusively.

If all ten characters in the S register and Q register compare equal, the compare word equal signal (CWEQ) will be generated from the JF module. Compare word equal allows comparison of the next pair of words.

If an inequality exists between the characters in S and Q, the respective compare character equal signals for the unequal characters will be at a zero level. These zero level compare character equal signals are monitored by an unequal character position priority encode circuit, located on the JF module. The output of the encoder provides a 4-bit binary code pointing to the first unequal character. This binary code is stored in the character position (CP) register.

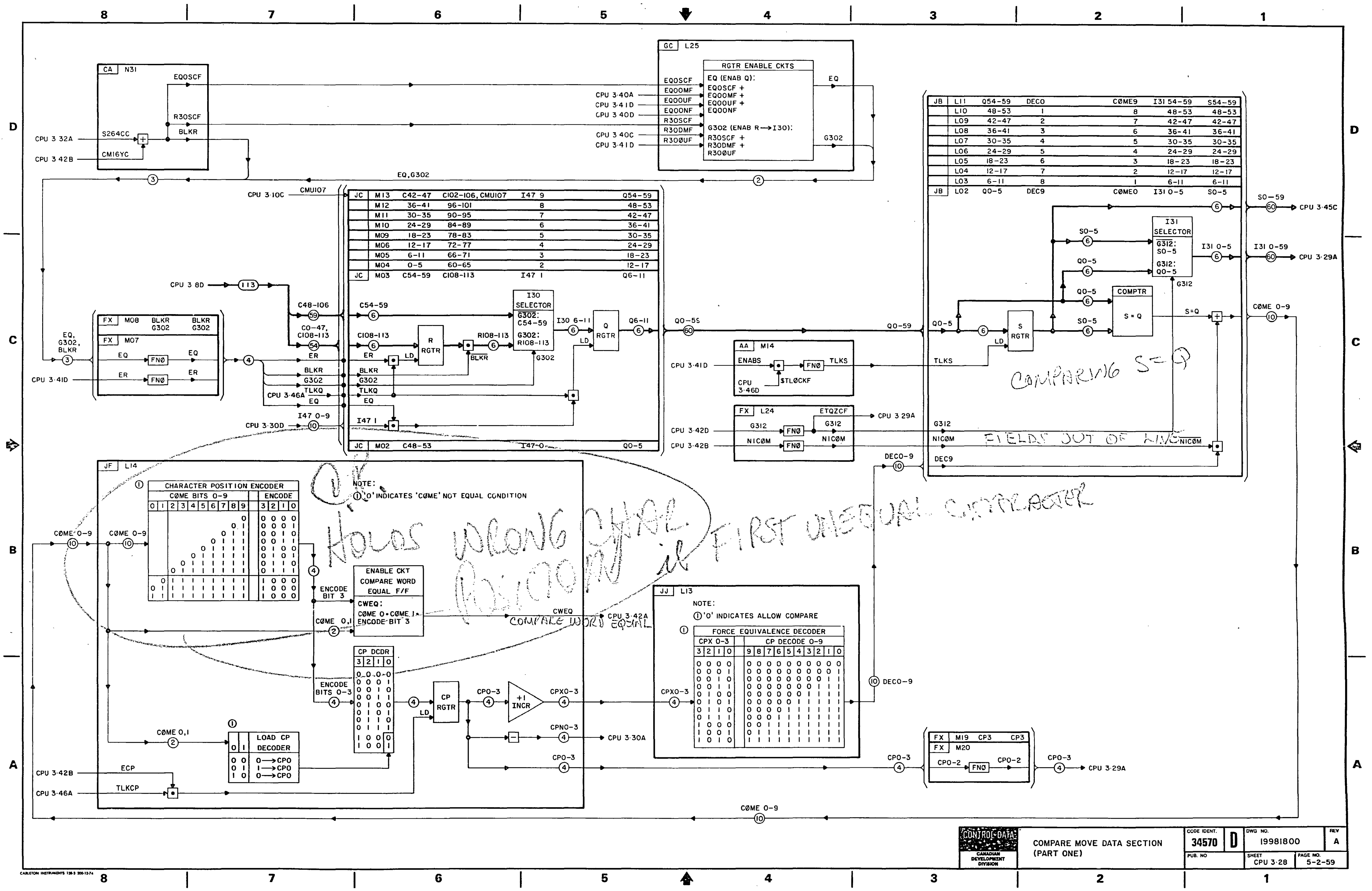
FORCE EQUIVALENCE CIRCUITS

The output of the character position register (CP) feeds a +1 incrementer circuit also located on the JF module. The incrementer is used to force equivalence for a collate instruction.

For example, assume that during the execution of a compare collate instruction a pair of characters are found unequal. The character position code for the unequal characters is stored in the character position register. The collating characters corresponding to the unequal characters are read from the collate table and compared. Should they be equal, the instruction continues. The code in the CP register is incremented by one, pointing to the next character to be compared. The incremented value is fed to the force equivalence decoder which generates 10 force equivalence bits (DEC 0-9). The force equivalence bits cause an equal comparison to occur on all characters preceding the unequal character and including the unequal character. Comparison of the remaining characters occurs as described previously.

TABLE 5-2-18. CPU 3.28 KEY TEST POINTS

	JC		JC		JB	
BIT NO.	PAK LOC.	C (IN)	PAK LOC.	Q REG (IN)	PAK LOC.	Q (IN)
00	M04	07	M02	02	L02	12
01	M04	05	M02	03	L02	13
02	M04	14	M02	04	L02	14
03	M04	12	M02	01	L02	10
04	M04	11	M02	10	L02	04
05	M04	13	M02	09	L02	09
06	M05	07	M03	02	L03	12
07	M05	05	M03	03	L03	13
08	M05	14	M03	04	L03	14
09	M05	12	M03	01	L03	10
10	M05	11	M03	10	L03	04
11	M05	13	M03	09	L03	09
12	M06	07	M04	02	L04	12
13	M06	05	M04	03	L04	13
14	M06	14	M04	04	L04	14
15	M06	12	M04	01	L04	10
16	M06	11	M04	10	L04	04
17	M06	13	M04	09	L04	09
18	M09	07	M05	02	L05	12
19	M09	05	M05	03	L05	13
20	M09	14	M05	04	L05	14
21	M09	12	M05	01	L05	10
22	M09	11	M05	10	L05	04
23	M09	13	M05	09	L05	09
24	M10	07	M06	02	L06	12
25	M10	05	M06	03	L06	13
26	M10	14	M06	04	L06	14
27	M10	12	M06	01	L06	10
28	M10	11	M06	10	L06	04
29	M10	13	M06	09	L06	09
30	M11	07	M09	02	L07	12
31	M11	05	M09	03	L07	13
32	M11	14	M09	04	L07	14
33	M11	12	M09	01	L07	10
34	M11	11	M09	10	L07	04
35	M11	13	M09	09	L07	09
36	M12	07	M10	02	L08	12
37	M12	05	M10	03	L08	13
38	M12	14	M10	04	L08	14
39	M12	12	M10	01	L08	10
40	M12	11	M10	10	L08	04
41	M12	13	M10	09	L08	09
42	M13	07	M11	02	L09	12
43	M13	05	M11	03	L09	13
44	M13	14	M11	04	L09	14
45	M13	12	M11	01	L09	10
46	M13	11	M11	10	L09	04
47	M13	13	M11	09	L09	09
48	M02		M12	02	L10	12
49	M02		M12	03	L10	13
50	M02		M12	04	L10	14
51	M02		M12	01	L10	10
52	M02		M12	10	L10	04
53	M02		M12	09	L10	09
54	M03		M13	02	L11	12
55	M03		M13	03	L11	13
56	M03		M13	04	L11	14
57	M03		M13	01	L11	10
58	M03		M13	10	L11	04
59	M03		M13	09	L11	09
108	M03	07				
109	M03	05				
110	M03	14				
111	M03	12				
112	M03	11				
113	M03	13				



CONTROL DATA
CANADIAN DEVELOPMENT DIVISION

COMPARE MOVE DATA SECTION (PART ONE)

CODE IDENT.		DWG NO.	REV
34570	D	19981800	A
PUB. NO		SHEET CPU 3·28	PAGE NO. 5-2-59

The circuitry shown on diagram 3.29 is used by the compare collate (466) and compare uncollated (467) instructions.

COMPARE UNCOLLATED (467)

For a compare uncollated instruction, the circuitry on this diagram determines whether the unequal character in Q was greater than, or less than, the unequal character in S. Selectors I32 and I33 on the JD module gate the unequal character from S via I37 to the TS register, and the unequal character from Q via I37 to the TQ register. The unequal character is selected using the code stored in the character position register (CP).

Each JD module is capable of performing a single bit comparison between TS and TQ. The output of the TS = TQ compare circuit generates a compare bit equal signal for each bit (CMTC 0-5). The compare bit equal signals will be at a one level when the respective bits in TS and TQ are equal.

The compare bit equal signals (CMTC 0-5) are fed to a priority encode circuit on the JE module. The priority encode circuit, scanning from left to right, produces a code pointing to the first unequal bit in the TS and TQ registers. The priority code is then fed to a multiplexer circuit. The multiplexer circuit monitors the TQ register bits 0-5. By using the binary code from the priority encoder, the multiplexer circuit will select the appropriate TQ register bit that compared unequal. If this bit equalled one, TQ would be greater than TS and the QGS signal will be generated.

QGS is used during the exit sequence to condition the X0 register.

COMPARE COLLATE (466)

For a compare collate instruction the circuitry on this diagram performs the collate operation.

I32 and I33 will select the unequal character from S and Q using the code stored in the CP register. These characters are then stored in the TS and TQ registers via the I37 selector.

Assuming that this is the first time a collate operation is being performed during the instruction execution, the T register will not contain a valid collate table word. Consequently, the word position register located on the JE module will not contain a valid word position code.

The word position register (WP) feeds two test circuits located on the JE module. These circuits determine whether the word position code is equal to the upper 3 bits of the unequal character in the TS and TQ registers. However, the output of these two test circuits are blocked by the first collate signal N1COL in its active state.

The collate sequence uses the output from the WP = TQ and the WP = TS test circuits to condition two equality detection FFs. The third equality detection FF is located on the JE module. A TQ = TS test circuit feeds the TQ = TS equality detection FF.

The contents of these three detection FFs determine the sequence of events that occur during a collate operation.

With the WP = TQ and WP = TS test circuits blocked by first collate active, only two equality detection possibilities can occur:

	TQ = TS	TQ = WP	TS = WP
1.	1	0	0
2.	0	0	0

The TQ = TS detection FF indicates that both collate characters are contained in the same collate table word. If $\overline{TQ = TS}$ the collate characters are located in different words of the collate table.

COLLATE TABLE LOOK-UP - $\overline{TQ = TS}$

Assuming TQ ≠ TS, two central memory references are required for collate table look-up. An address pointing to the first word of the collate table is stored in the A0 register. The contents of A0 are added to TS register bits 3-5 to provide the address for the first table look-up. TS register bits 3-5 are also gated to the WP register via I35.

After the word read from the collate table is received at the CR9 register, it is gated to the table register (T) located on the JG module. The table register feeds a priority multiplex circuit capable of selecting one of the eight collate characters from table register. Selection is determined by the binary code selected via I38 located on the JD module. At this time I38 would select the lower 3 bits of the TS register (0-2) to I38, so that the appropriate character is selected in I34. The selected collate character is gated to I37 located on the JD module. I37 will now select the collate character from I34 to be stored in the TS register.

Another memory reference will obtain the second word from the collate table. The procedure is similar to that already described except that the TQ register is used.

After the second word read from the collate table is received at the CR9 register, it is gated to the table register, destroying the previous contents. The lower 3 bits (0-2) of the TQ register are gated to I38, allowing selection of the second collate character from I34 to I37. I37 will select the collate character to be stored in the TQ register.

With both collate characters stored in the TS and TQ registers, the contents of TS and TQ are compared for equality using the comparison circuit located on the JD module. If both collate characters are equal, the collate character equal signal (CCEQ) from the JE module will allow normal continuance of instruction execution.

COLLATE CHARACTER COMPARISON - EQUAL

The remaining circuits on the JE module serve a useful purpose if the result of a collate character comparison is equal. The word position register (WP) will contain a code pointing to the collate table word referenced on the last memory request. This is the word contained in the table register.

If during the same collate instruction execution another collate operation is required, the collate sequence control logic will check the condition of the three equality detection FFs to determine the sequence of events.

Five possible combinations can occur: BOTH CHARACTERS IN LAST TABLE LOADED

	TQ = TS	TQ = WP	TS = WP	MEMORY REQUESTS
1.	-	1	1	NONE
2.	0	0	0	2
3.	1	0	0	1
4.	-	0	1	1
5.	-	1	0	1

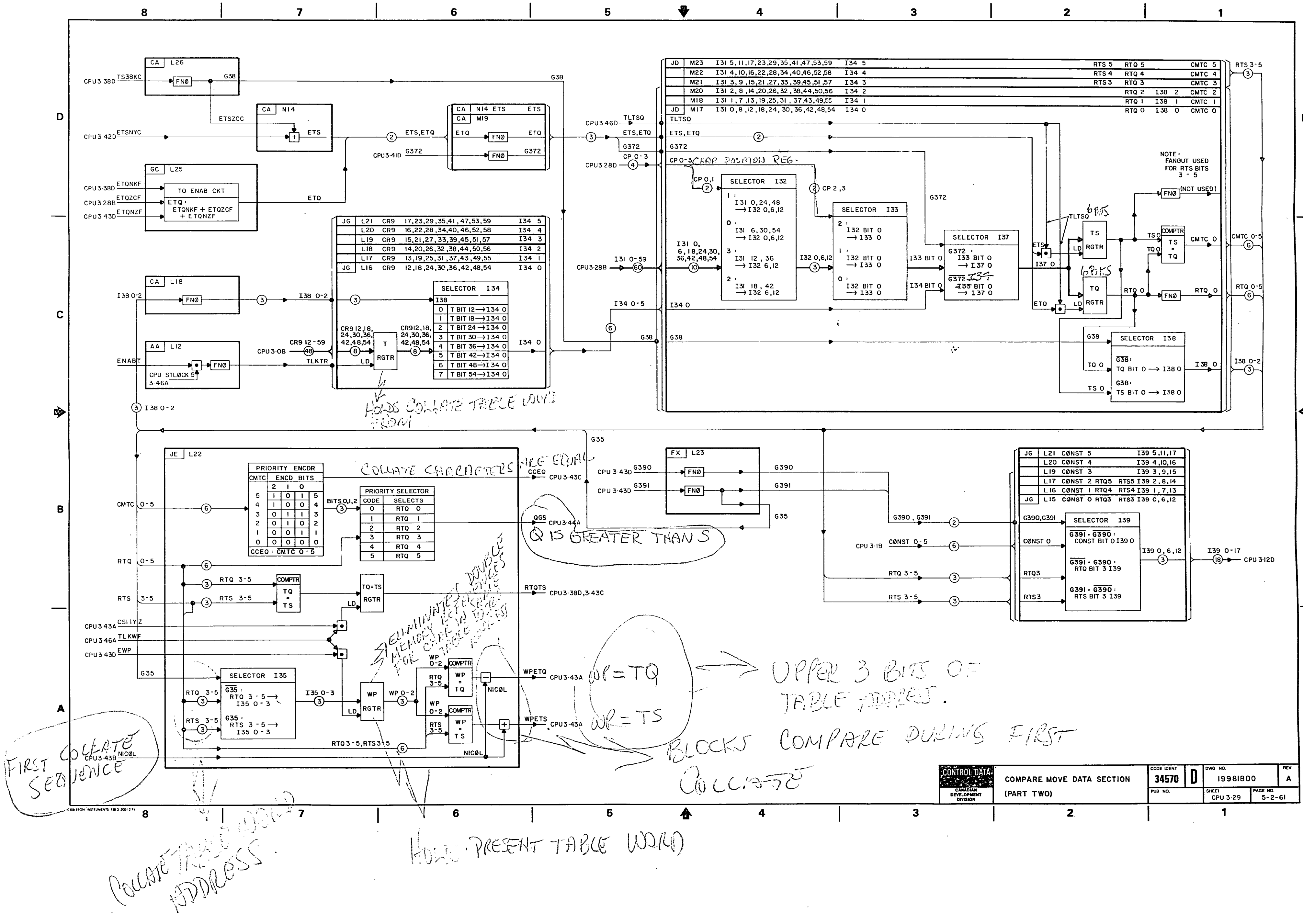
If both TQ = WP and TS = WP, both collate characters are stored in the table register; a memory reference is not required.

If $\overline{TQ = TS}$ and $\overline{TQ = WP}$ and $\overline{TS = WP}$, neither collate character is stored in the table register. Two memory requests are required to obtain the collate characters from the collate table.

Finally, if either TQ = WP or TS = WP, both collate characters are located in the same table word. Only one memory request is required to obtain both characters from the table.

TABLE 5-2-19. CPU 3.29 KEY TEST POINTS

	FG		JD	
BIT NO.	PAK LOC.	CR9 (IN)	PAK LOC.	I31 (IN)
00			M16	03
01			M17	03
02			M18	03
03			M21	03
04			M22	03
05			M23	03
06			M16	06
07			M17	06
08			M18	06
09			M21	06
10			M22	06
11			M23	06
12	L15	07	M16	04
13	L16	07	M17	04
14	L17	07	M18	04
15	L19	07	M21	04
16	L20	07	M22	04
17	L21	07	M23	04
18	L15		M16	13
19	L16		M17	13
20	L17		M18	13
21	L19		M21	13
22	L20		M22	13
23	L21		M23	13
24	L15	05	M16	12
25	L16	05	M17	12
26	L17	05	M18	12
27	L19	05	M21	12
28	L20	05	M22	12
29	L21	05	M23	12
30	L15	03	M16	07
31	L16	03	M17	07
32	L17	03	M18	07
33	L19	03	M21	07
34	L20	03	M22	07
35	L21	03	M23	07
36	L15	09	M16	05
37	L16	09	M17	05
38	L17	09	M18	05
39	L19	09	M21	05
40	L20	09	M22	05
41	L21	09	M23	05
42	L15	08	M16	11
43	L16	08	M17	11
44	L17	08	M18	11
45	L19	08	M21	11
46	L20	08	M22	11
47	L21	08	M23	11
48	L15	14	M16	14
49	L16	14	M17	14
50	L17	14	M18	14
51	L19	14	M21	14
52	L20	14	M22	14
53	L21	14	M23	14
54	L15	13	M16	02
55	L16	13	M17	02
56	L17	13	M18	02
57	L19	13	M21	02
58	L20	13	M22	02
59	L21	13	M23	02



CONTROL DATA
CANADIAN DEVELOPMENT DIVISION

COMPARE MOVE DATA SECTION
(PART TWO)

CODE IDENT		DWG NO.	REV
34570	D	19981800	A
PUB NO.		SHEET CPU 3-29	PAGE NO. 5-2-61

C1, C2 OFFSET REGISTERS

The C1 and C2 offset registers are located on the JX module. C1 provides a 4-bit offset value for the first word of the K1 field. C2 provides a 4-bit offset value for the first word of the K2 field.

I41, I42 - C-ADDER

Selector circuits I41 and I42 are located on the JX module. The outputs of I41 and I42 provide the A and B inputs to the C adder on the JY module. Depending upon the gating term selection of I41 and I42 (both operate in parallel), the C adder may perform three functions:

1. Subtract C2 from C1 (by complement addition)
2. Subtract C1 from C2 (by complement addition)
3. Add SCR+ ($+12_8$)

The C adder output is gated to the shift count register (SCR). The shift count value is used to shift characters in the C register to the appropriate position (depending on the C1, C2 offset value) before loading in the Q register. The shift count value stored in SCR represents the number of characters that must be right shifted. This value is gated to a times-six translator circuit that converts the character shift count to a bit shift count. The translator output (SCRX 1-5) is gated to the shift count register via I19 and I9 (CPU 2.8).

I40, I40 DECODE, I44

Selector I40 located on the JX module provides a 4-bit input path to selector I44 and the I40 decode circuit located on the JJ module. Depending on gating term selection of I40, the contents of any one of four registers may be gated to the character select register (CSR) located on the JZ module.

$$\begin{tikzcd}
C1 \arrow[r] & I40 \arrow[r] & \text{I40 DECODE} \arrow[r] & CSR \\
C2 \arrow[r] & I40 \arrow[r] & \text{I40 DECODE} \arrow[r] & CSR \\
LA \arrow[r] & I40 \arrow[r] & \text{I40 DECODE} \arrow[r] & CSR \\
LC \arrow[r] & I40 \arrow[r] & \text{I40 DECODE} \arrow[r] & CSR
\end{tikzcd}$$

In the same manner, the gating term selection of I40 allows the contents of any one of three registers to be gated to the partial write register (PW) also located on the JZ module.

$$\begin{tikzcd}
SCR \arrow[r] & I40 \arrow[r] & \text{I40 DECODE} \arrow[r] & PW \\
LA \arrow[r] & I40 \arrow[r] & \text{I40 DECODE} \arrow[r] & PW \\
LC \arrow[r] & I40 \arrow[r] & \text{I40 DECODE} \arrow[r] & PW
\end{tikzcd}$$

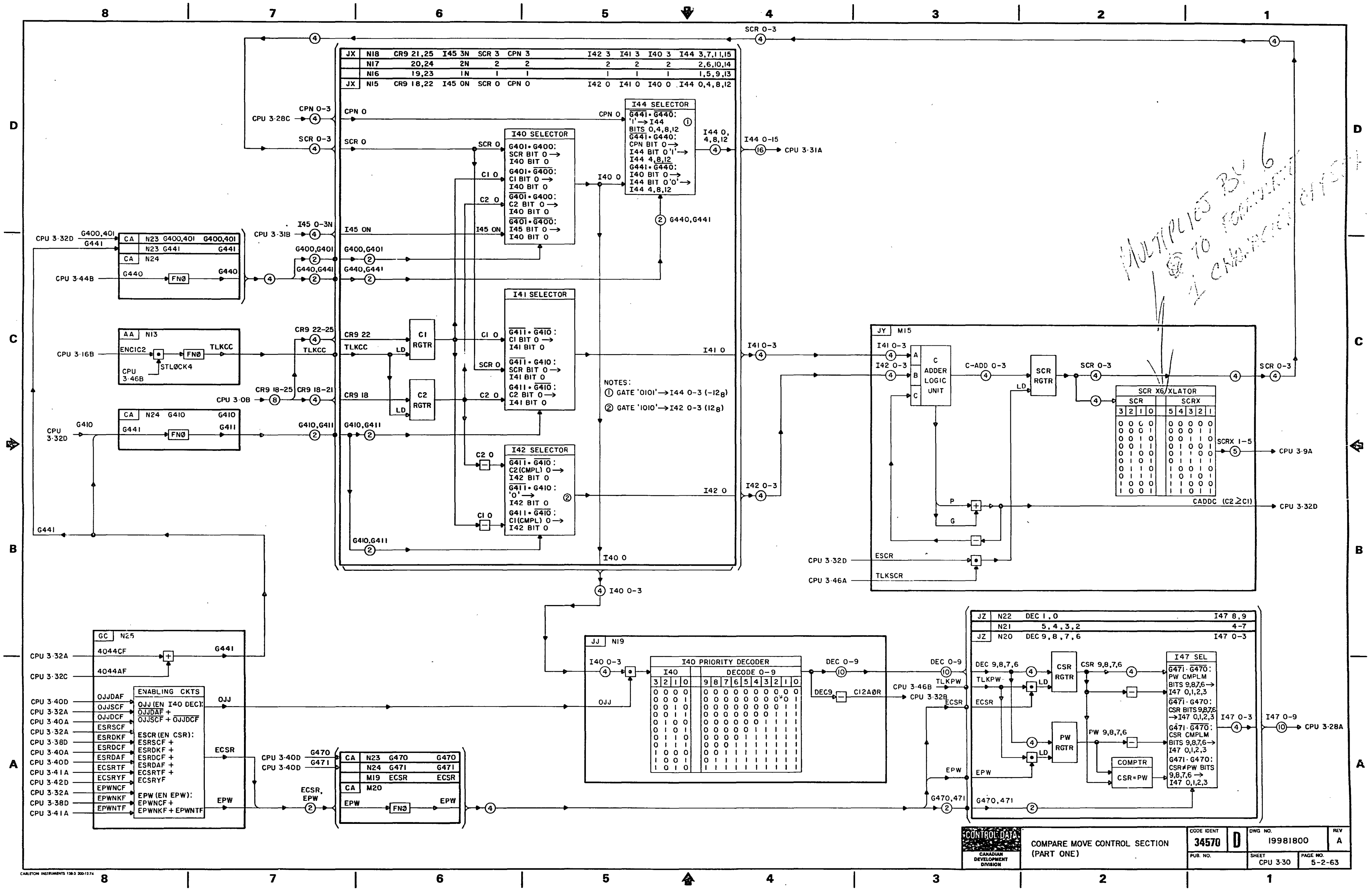
The purpose of the I40 decoder is to transform the 4-bit code from one of the registers listed above to a 10-bit code representing the ten character positions in a word.

Selector I44 located on the JX module provides a 16-bit input path to the LE register located on the JM module (CPU 3.31). The gating term selection of I44 allows the contents of any one of three registers, or a generated constant value to be gated to the LE register.

$$\begin{tikzcd}
C1 \arrow[r] & I40 \arrow[r] & I44 \arrow[r] & LE \\
C2 \arrow[r] & I40 \arrow[r] & I44 \arrow[r] & LE \\
 & \overline{CP} \arrow[r] & I44 \arrow[r] & LE \\
 & -12_8 \arrow[r] & I44 \arrow[r] & LE
\end{tikzcd}$$

CHARACTER SELECT/PARTIAL WRITE REGISTERS (CSR, PW)

The character select and partial write registers contain a 10-bit code, each bit representing one of ten character positions in the Q register. CSR, CSR complement, PW complement and CSR $\neq$ PW are gated to the I47 selector. Depending on gating term selection of I47, the appropriate CSR/PW bits provide an enable or disable on the Q register input load circuit (CPU 3.28).



CONTROL DATA
CANADIAN DEVELOPMENT DIVISION

COMPARE MOVE CONTROL SECTION
(PART ONE)

CODE IDENT		DWG NO.	REV
34570	D	19981800	A
PUB. NO.		SHEET CPU 3-30	PAGE NO. 5-2-63

DETAILED PAK DIAGRAM (CPU 3.31)

COMPARE/MOVE CONTROL SECTION (Part Two)

LA, LC REGISTERS

The LA and LC registers are located on the JN module. At the beginning of a compare/ move instruction, the LA and LC registers will contain an octal representation of the character field length. The length value is gated from CR9 bits 26-29, 48-50 (465, 466, 467) or CR9 bits 26-29, 48-56 (464), via the I46 selector located on the JM module.

LAC1, LAC2 REGISTERS

The LAC1 and LAC2 registers receive the current LA or LC length value via selector I45. The length value in LAC2 is used during compare unequal to determine the character count value to be stored in the X0 register upon instruction conclusion.

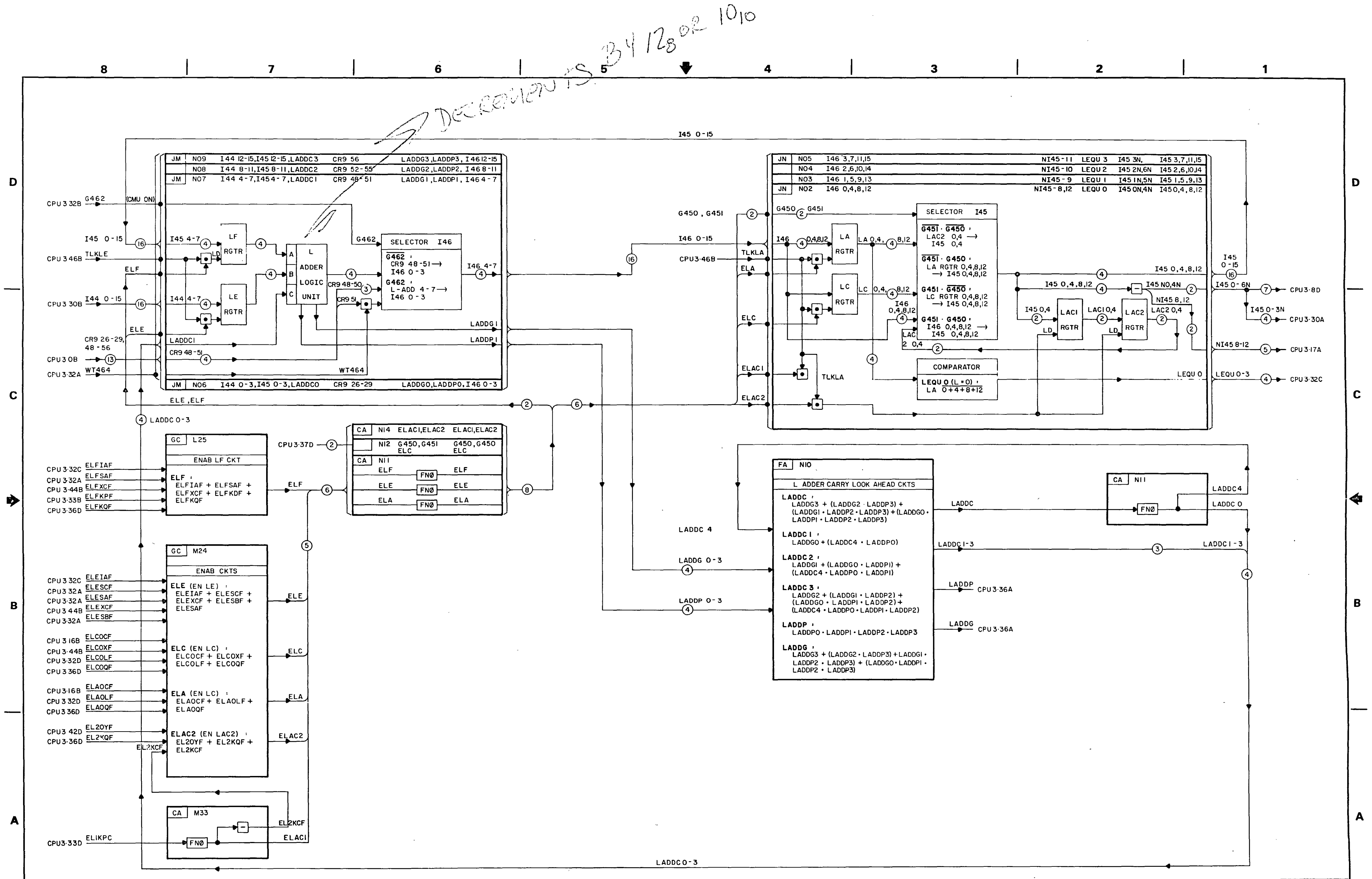
LE, LF REGISTERS

The LE and LF registers provide the A and B input path to the L adder located on the JM module. The LE register receives its input from selector I44 located on the JX module (CPU 3.30). Depending on the selections made at I40 and I44 (described previously), the LE register will receive either the contents of C1 or C2, the complemented contents of the character position (CP) register, or a generated constant value of -12_8.

The LF register receives its input from selector I45. I45 allows the contents of LA, LC, LAC2 or the L adder output via I46 to be gated to the LF register.

L ADDER

The L adder performs four functions required for character length calculations. Initially, the contents of LA and LC are added to the contents of C1 and C2, respectively. The results are returned to the LA and LC registers. During K1 and K2 address sequences, LA and LC are decremented by -12_8. The address sequence monitors the decremented values of LA and LC to determine a K1 or K2 exhaust condition. In addition, during K2 address sequences for a move, the decremented LC value from the L adder is gated via I45 to the LF register to perform a double subtraction. The double subtraction provides a look-ahead function that detects an exhaust condition on the next K2 address sequence. Finally, the L adder allows the contents of the character position register (CP) to be subtracted from the decremented LA or LC value in the LAC2 register on compare unequal. The resultant value will be stored in the X0 register upon instruction termination.



DECREMENTS B4 12₈ or 10₁₀
I45 0-15
JM N09 I44 12-15, I45 12-15, LADDC3 CR9 56 LADDG3, LADDP3, I46 12-15
N08 I44 8-11, I45 8-11, LADDC2 CR9 52-55 LADDG2, LADDP2, I46 8-11
JM N07 I44 4-7, I45 4-7, LADDC1 CR9 48-51 LADDG1, LADDP1, I46 4-7
JM N06 I44 0-3, I45 0-3, LADDC0 CR9 26-29 LADDG0, LADDP0, I46 0-3
CPU 3.32B G462 (CMU ON)
I45 0-15
CPU 3.46B TLKLE
CPU 3.30B I44 0-15
CR9 26-29, 48-56
CPU 3.0B
CPU 3.32A WT464
LF RGTR
LE RGTR
L ADDER LOGIC UNIT
SELECTOR I46
G462 · CR9 48-51 → I46 0-3
G462 · L-ADD 4-7 → I46 0-3
I46 4-7
LADDG1
LADDP1
LADDC1
CR9 48-51
WT464
ELE, ELF
LADDC 0-3
JN N05 I46 3,7,11,15 NI45-11 LEQU 3 I45 3N, I45 3,7,11,15
N04 I46 2,6,10,14 NI45-10 LEQU 2 I45 2N,6N I45 2,6,10,14
N03 I46 1,5,9,13 NI45-9 LEQU 1 I45 1N,5N I45 1,5,9,13
JN N02 I46 0,4,8,12 NI45-8,12 LEQU 0 I45 0N,4N I45 0,4,8,12
G450, G451
I46 0-15
CPU3-46B
TLKLA
LA RGTR
LC RGTR
SELECTOR I45
G451 · G450 · LAC2 0,4 → I45 0,4
G451 · G450 · LA RGTR 0,4,8,12 → I45 0,4,8,12
G451 · G450 · LC RGTR 0,4,8,12 → I45 0,4,8,12
G451 · G450 · I46 0,4,8,12 → I45 0,4,8,12
COMPARATOR
LEQU 0 (L=0) · LA 0+4+8+12
LAC1 RGTR
LAC2 RGTR
I45 0-6N → CPU3-8D
I45 0-3N → CPU3-30A
NI45 8-12 → CPU3-17A
LEQU 0-3 → CPU3-32C
CPU3-37D
CA NI4 ELAC1, ELAC2 ELAC1, ELAC2
NI2 G450, G451 ELC G450, G450 ELC
CA NI1 ELF FNØ ELF
ELE FNØ ELE
ELA FNØ ELA
GC L25 ENAB LF CKT
ELF · ELFIAF + ELFSAF + ELFXCF + ELFKDF + ELFKQF
CPU 3.32C ELFIAF
CPU 3.32A ELFSAF
CPU 3.44B ELFXCF
CPU 3.33B ELFKPF
CPU 3.36D ELFKQF
GC M24 ENAB CKTS
ELE (EN LE) · ELEIAF + ELESCF + ELEXCF + ELESBF + ELESAF
ELC (EN LC) · ELCOCF + ELCOXF + ELCOLF + ELCOQF
ELA (EN LC) · ELAOCF + ELAOLF + ELAOQF
ELAC2 (EN LAC2) · EL2OYF + EL2KQF + EL2KCF
CPU 3.32C ELEIAF
CPU 3.32A ELESCF
CPU 3.32A ELESAF
CPU 3.44B ELEXCF
CPU 3.32A ELESBF
CPU 3.16B ELCOCF
CPU 3.44B ELCOXF
CPU 3.32D ELCOLF
CPU 3.36D ELCOQF
CPU 3.16B ELAOCF
CPU 3.32D ELAOLF
CPU 3.36D ELAOQF
CPU 3.42D EL2OYF
CPU 3.36D EL2KQF
CA M33
CPU3-33D ELIKPC FNØ
EL2KCF
ELAC1
FA N10 L ADDER CARRY LOOK AHEAD CKTS
LADDC · LADDG3 + (LADDG2 · LADDP3) + (LADDG1 · LADDP2 · LADDP3) + (LADDG0 · LADDP1 · LADDP2 · LADDP3)
LADDC1 · LADDG0 + (LADDC4 · LADDP0)
LADDC2 · LADDG1 + (LADDG0 · LADDP1) + (LADDC4 · LADDP0 · LADDP1)
LADDC3 · LADDG2 + (LADDG1 · LADDP2) + (LADDG0 · LADDP1 · LADDP2) + (LADDC4 · LADDP0 · LADDP1 · LADDP2)
LADDP · LADDP0 · LADDP1 · LADDP2 · LADDP3
LADDG · LADDG3 + (LADDG2 · LADDP3) + LADDG1 · LADDP2 · LADDP3) + (LADDG0 · LADDP1 · LADDP2 · LADDP3)
LADDC 4
LADDG 0-3
LADDP 0-3
LADDC
LADDC 1-3
LADDP CPU 3.36A
LADDG CPU 3.36A
CA NI1 FNØ LADDC4 LADDC 0
LADDC 0-3

CONTROL DATA
CANADIAN DEVELOPMENT DIVISION

COMPARE MOVE CONTROL SECTION
(PART TWO)

CODE IDENT		DWG NO.	REV
34570	D	19981800	A
PUB NO		SHEET CPU 3-31	PAGE NO 5-2-65

CARLETON INSTRUMENTS 138-3 200-1274

DETAILED PAK DIAGRAM (CPU 3.32)

INSTRUCTION DECODE SEQUENCE, START SEQUENCE

The instruction decode sequence is initiated from the common time sequence by the GOCMU signal. The sequence decodes a 460 (pass), 464 (move indirect), 465 (move direct), 466 (compare collate), 467 (compare uncollate).

A decode of 460-463 for a pass instruction will generate the NOP signal. NOP enables the RNI sequence.

A decode of 464 for move indirect generates the enable increment sequence signal (EINCS). The increment sequence will use bit positions 30-50 of the instruction word to address (Bj) + K, which will be the address of a 60-bit descriptor word. On receipt of the descriptor word from memory, the accept sequence will generate a GO464 signal to initiate the instruction decode sequence once again.

GOCMU for a 465, 466 or 467, or GO464, and the character length value not equal zero will generate the enable start sequence signal (ESTAHL).

MOVE INSTRUCTION (464, 465 - Refer to timing diagram, figure 5-2-31)

During the instruction decode and start sequence for a move, the following will occur:

1. C2 offset plus character length value in LC are added in L adder. Result returned to LC.
2. C1 is subtracted from C2 (by complement addition); the result is stored in the shift count register (SCR). The output of the C adder is monitored by the C2 $\geq$ C1 FF located on the JL module. If C2 $\geq$ C1, a carry signal (CADDC) will enable setting the C2 $\geq$ C1 FF.
3. C1 and C2 are tested for out of range condition by the I40 decode circuit. The C12AOR signal will be generated if C1 $\geq 10_{10}$ or C2 $\geq 10_{10}$. C12AOR will set the C1/C2 AOR FF located on the JL module.
4. If C1 > C2 (that is C2 $\geq$ C1 FF reset), + 12_8 is added to the shift count value in the SCR register.
5. The shift count value in SCR is gated to a times-six decoder circuit, then to I19, I9 and the SK register.
6. C1 offset plus character length in LA are added in L adder. Result returned to LA.
7. SCR register shift count value is gated to the I40 decoder and stored in the partial write register for use by the data sequences.
8. -12_8 generated at I44 is stored in the LE register for subtraction during the address sequences.

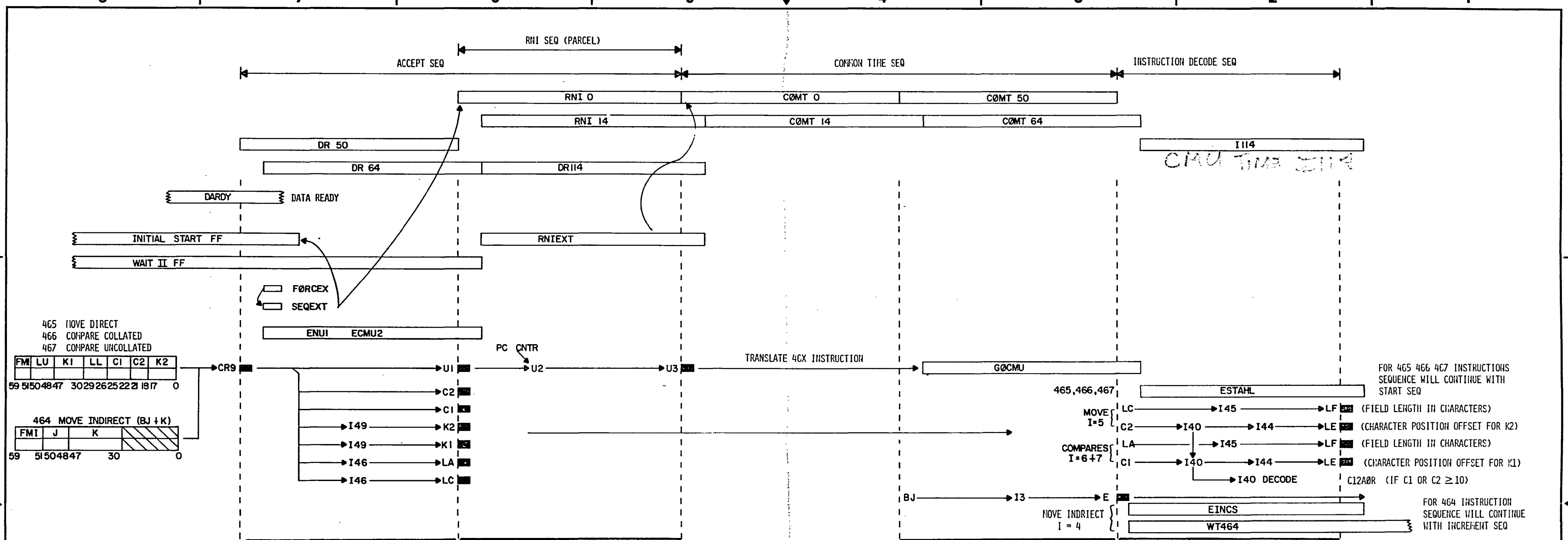
COMPARE INSTRUCTION (466, 467 - Refer to timing diagram, figure 5-2-31)

The instruction decode and start sequence is similar to that of a move instruction except for the following:

1. The addition of C1 + LA occurs before the addition of C2 + LC. This is because the address sequence addresses the K2 word first for a move and K1 word first for a compare.
2. If C1 > C2, as determined by step 2 above, the sequence will not add + 12_8 to the SCR register (as in step 4); it will, however, subtract C2 from C1 to obtain a new shift count value in SCR.

The start sequence initiates the address sequence by generating the clear block K address FF signal (CBKOCF). The address sequence operates in parallel with the start sequence starting at S164 time.

INST. IS XLATED IN C.P.U.



TERM NAME	COMMAND OR FUNCTION	DPD NO
DR64	(INTSRT:R1WTII)	16
FORCEX	FORCE EXIT	14
SEQEXT	SEQUENCE EXIT	
DR64	(NRWTII)	16
ENU1	CR9 → U1	1
ECMUR		16
ENC1C2	$CR9_{18-25} \rightarrow$ C1,C2	30
$\overline{CM00}$		32
$\overline{G462}$	$CR9_{26-29,\ 48-50} \rightarrow$ I46	31
$\overrightarrow{G462}$	$CR9_{30-47,\ 0-17} \rightarrow$ I49	3
ECMUR		
ELAOCF		16
ELCOCF		16
ELA	$I46_{0-15} \rightarrow$ LA,LC	31
ELC		31
ECMUR		
EK1OCF		16
EK2OCF		16
EK1	$I49_{30-47} \rightarrow$ K1	3
EK2	$I49_{0-17} \rightarrow$ K2	3

TERM NAME	COMMAND OR FUNCTION	DPD NO
RNI TO		14
U2U3	U2 → U3	1
RNI T14	(RUN FF SET BY PREVIOUS RNI SEQ)	14
	RNIEXT ENABLES COMMON TIME SEQ	17

TERM NAME	COMMAND OR FUNCTION	DPD NO
CØM50		15
SELBJ	ADRS BJ RGTR	2
CØMT64	46X	15
COMBI3		15
SELBI3	BJ → I3	12
NCØ50		15
ENABE	I3 → E	13

TERM NAME	COMMAND OR FUNCTION	DPD NO
I114		32
LA45HV	(I=6+7-COMPARE)	
G450		37
$G450 \cdot \overline{G451}$	LA → I45	31
$\overline{G450} \cdot G451$	LC → I45	31
I114		32
ELFIAF		
ELF	I45 → LF	31
C140HL	(I=6+7-COMPARE)	32
G401		
C240HL	(L=5-MOVE)	32
G400		
$G400 \cdot \overline{G401}$	C2 → I40	30
$\overline{G400} \cdot G401$	C1 → I40	30
I114		32
4044HF		
G441		30
$\overline{G440} \cdot G441$	I40 → I44	30
I114		32
ELEIAF		
ELE	I44 → LE	31
ESTAHL	EN START SEQ (PCEQ0·L≠0·465+466+467)	32

OJJ	TEST I40Z10	30
C12AØR	ENABLE AØR INHIBIT START SEQ T164	32
EINCS	(GØCMU·I=4)	
WT464	SET WT464 FF	32

CONTINUED

(Part 1 of 3)

CONTROL DATA CANADIAN DEVELOPMENT DIVISION

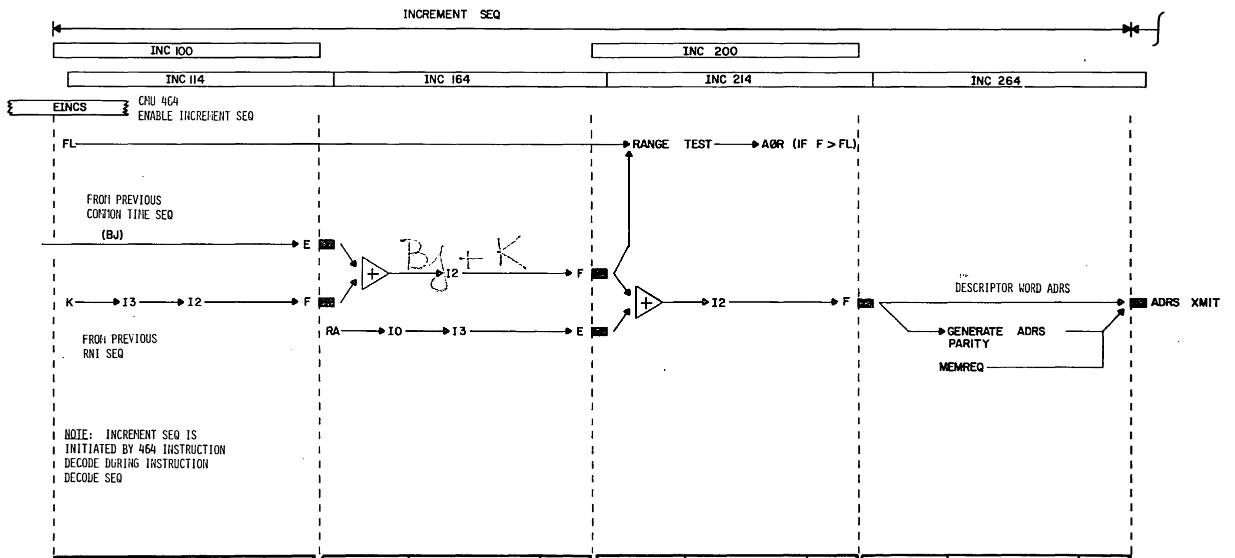
INSTRUCTION FLOW INSTRUCTION DECODE SEQ

CODE IDENT.		DWG. NO.	REV
34570	D	19981800	A
FIGURE 5-2-30		PAGE NO. 5-2-66.2	

CARLETON INSTRUMENTS 139-3 300-12-74

CPU

CMU



NOTE: INCREMENT SEQ IS INITIATED BY 464 INSTRUCTION DECODE DURING INSTRUCTION DECODE SEQ

TERM NAME	COMMAND OR FUNCTION	DPD NO
INC114		21
IN114B	(NF46X)	
SELKI3	K ⟶ I3	13
INC114		21
NI114		
I3I2	I3 ⟶ I2	13
ENABF	I2 ⟶ F	13

TERM NAME	COMMAND OR FUNCTION	DPD NO
$\overline{SI00}\cdot\overline{SI01}\cdot\overline{SI02}$	RA ⟶ IØ	3
INC164		21
NI164	IØ ⟶ I3	12
	I3 ⟶ E	13
$\overline{I3I2}$	F ADD ⟶ I2	13
INC164		21
NINX64	I2 ⟶ F	13
ENABF		

TERM NAME	COMMAND OR FUNCTION	DPD NO
INC214		21
NI214C	TEST AØR	
ENABØR	F > FL	17
$\overline{I3I2}$	F ADD ⟶ I2	13
INC214		21
NINX64		
ENABF	I2 ⟶ F	13

TERM NAME	COMMAND OR FUNCTION	DPD NO
INC264		21
NIN264		
CEMREQ		17
CEMREQ		17
ENBADT	ENABLE ADRS	17
TLKAD	XMITTERS	45
MEMREQ	REQUEST CENTRAL MEMORY	17

C.P.U.

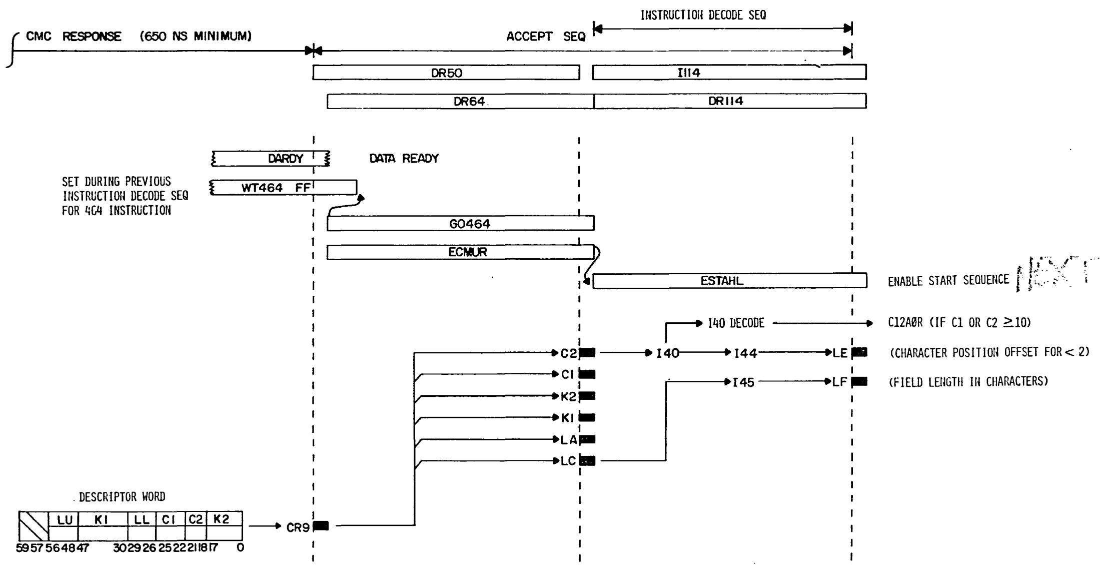
(Part 2 of 3)

CONTROL DATA
CANADIAN DEVELOPMENT DIVISION

INSTRUCTION FLOW
INSTRUCTION : DECODE SEQ

CODE IDENT.		DWG. NO.	REV
34570	D	19981800	A
FIGURE 5-2-30		PAGE NO. 5-2-66-3	

CARLETON INSTRUMENTS



NEXT PAGE

TERM NAME	COMMAND OR FUNCTION	DPD NO
DR64		16
G0464	(WT464 FF)	
	RESET WT464 FF	32
GØ464		
ECMOR		16
ENC1C1	$CR9_{18-25} \rightarrow$ C1,C2	30
ELA	$CR9_{26-29} \rightarrow$ LA,LC	31
ELC	48-56	
EK1	$CR9_{30-47} \rightarrow$ K1	3
EK2	$CR9_{0-17} \rightarrow$ K1	3

TERM NAME	COMMAND OR FUNCTION	DPD NO
GØ464	ENABLE INSTRUCTION DECODE SEQ	32
I114		32
$\overline{\text{LA45HV}}$		
G451		
$\overline{\text{G450}}$·G451	LC $\rightarrow$ I45	31
I114		32
ELFIAF		
ELF	I45 $\rightarrow$ LF	31
C240HL		
G400		
G400·$\overline{\text{G401}}$	C2 $\rightarrow$ I40 (AØR)	30
I114		32
4044AF		
G441		30
$\overline{\text{G440}}$·G441	I40 $\rightarrow$ I44	30
I114		32
ELEIAF		
ELE	I44 $\rightarrow$ LE	31
ESTAHL	EN START SEQ ($\overline{\text{AØ}}$R)	32
OJJ	TEST I40 Z10	30
C1C2AØR	ENABLE AØR INHIBIT START SEQ T164	32

(Part 3 of 3)

CONTROL DATA CANADIAN DEVELOPMENT DIVISION	INSTRUCTION FLOW INSTRUCTION DECODE	CODE IDENT 34570	D	DWG. NO. 19981800	REV A
		FIGURE 5-2-30			PAGE NO 5-2-66·4

	I114	S114	S164 (K114)	S214 (K164)	S264 (K214)
		C1 → I40 DECODE → C1 ≥ 10 -12_8 → I44 → LE C2 → I41, $\overline{C1}$ → I42 → + → SCR (CADDC) → C2 ≥ C1	C2 → I40 DECODE → C2 ≥ 10	0 → I40 DECODE → CSR SCR → I19 → I9 → SK	12_8 → I44 → LE SCR → I40 DECODE → PW
MOVE INSTRUCTION START SEQ FOR MOVE	LC → I45 → LF C2 → I40 → I44 → LE	LF, LE → + → I46 → LC	SCR → I41, $+12_8$ → I42 → + → SCR (IF C1 > C2)	LA → I45 → LF C1 → I40 → I44 → LE	LF, LE → + → I46 → LA
COMPARE INSTRUCTION START SEQ FOR COMP.	LA → I45 → LF C1 → I40 → I44 → LE	LF, LE → + → I46 → LA	C1 → I41, $\overline{C2}$ → I42 → + → SCR (IF C1 > C2)	LC → I45 → LF C2 → I40 → I44 → LE	LF, LE → + → I46 → LC
	CMU EXIT TO RNI IF GØ464. AØRI		CMU EXIT TO RNI IF C1 ≥ 10	CMU EXIT TO RNI IF C2 ≥ 10	

CONTROL DATA
CANADIAN DEVELOPMENT DIVISION

INSTRUCTION DECODE SEQ
START SEQ

CODE IDENT		DWG NO	REV
34570	D	19981800	A
FIGURE 5-2-31		PAGE NO 5-2-66-5	

TABLE 5-2-20. COMPARE/MOVE COMMAND TIMING

SEQUENCE: INSTRUCTION DECODE

TIME	SIGNAL NAME	TEST POINT	P	COMMAND	CONDITION	COMMENTS
	(3.32)			ENABLE I114	GOCMU + G0464	
I114						
	NOP (3.32)			ENABLE NO OPERATION	GOCMU . [$\overline{464}+\overline{465}+\overline{466}+\overline{467}$]	
	NILLI (3.32)			ENABLE ILLEGAL INSTRUCTION	GOCMU . [$\overline{464}+\overline{465}+\overline{466}+\overline{467}.\overline{PC=0}$]	
	NCMUEX (3.32)			ENABLE CMU EXIT	GO464 . AORMQH + EMCEXH	
	CMUCHC (3.32)	L32-9	F	ENABLE CMU MASTER CLR	NCMUEX + NMCL + EMCEXH	
	ESTAHL (3.32)	M26-8	T	ENABLE START SEQ	GOCMU . [(465+466+467) + GO464 . AORNQH $L \neq 0$	
	EINCS (3.32)			ENABLE INCREMENT SEQ	GOCMU . 464 . PCEQ1	
	(3.32)			CLR WAIT 464 FF	GO464 + NMCL	
	I114HA (3.32)	M33-11	F	ENABLE LE RGTR	GOCMU + GO464	
	I114HA (3.32)	M33-11	F	ENABLE LF RGTR	GOCMU + GO464	
	I114HA (3.32)	M33-11	F	SELECT I40 → I44	GOCMU + GO464	
	C240HL (3.32)	M26-5	F	SELECT C2 → I40	GOCMU . 465 PCEQ1 + G0464. $\overline{AORNQH}$	
	C140HL (3.32)	M26-4	F	SELECT C1 → I40	GOCMU . (466+467)	
	NCMULO (3.32)			ENABLE 0 → X EXIT	GOCMU . (465+466+467) . L = 0	

TABLE 5-2-21. COMPARE/MOVE COMMAND TIMING

SEQUENCE: START

Page #1 of 2

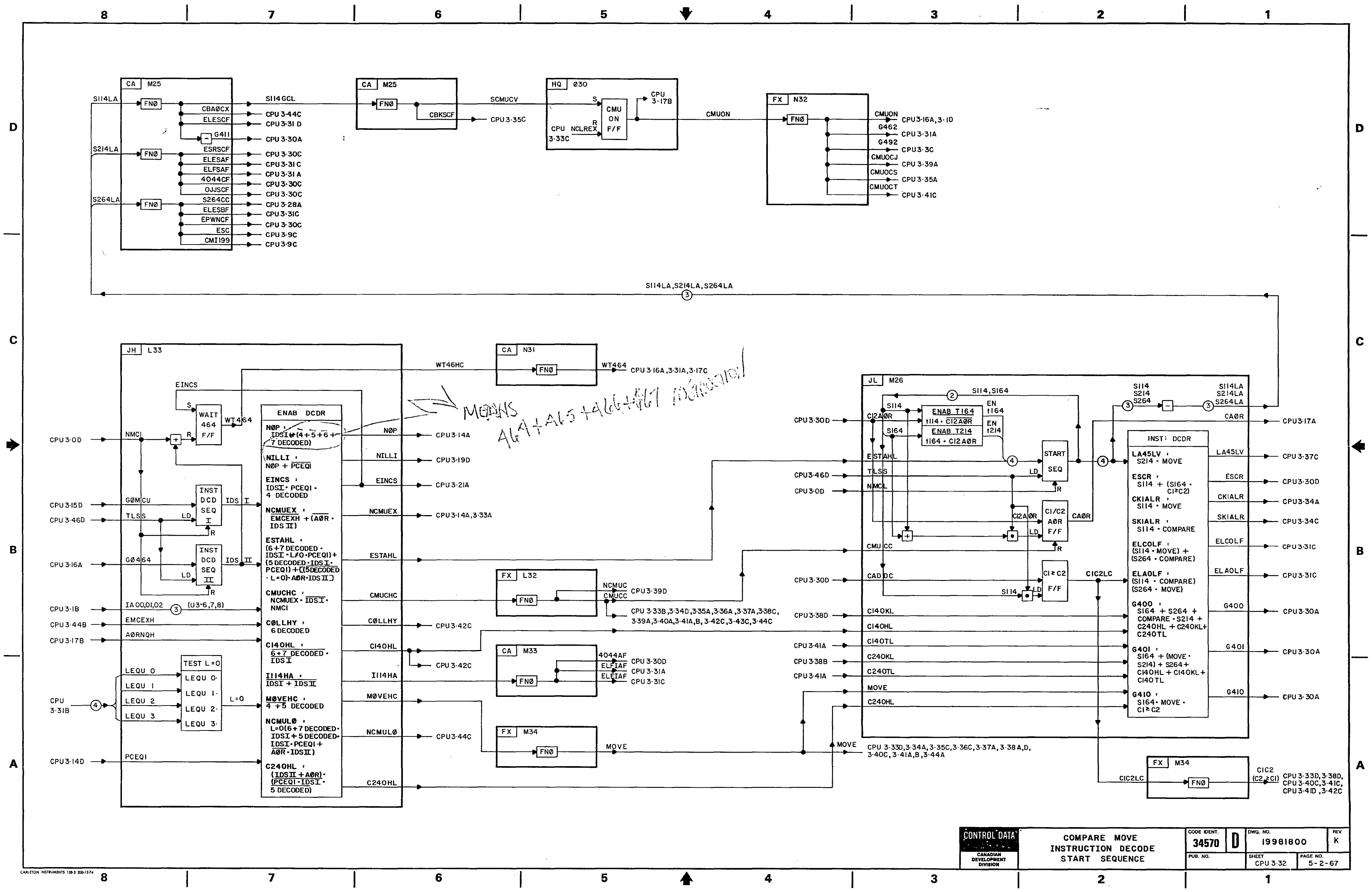
TIME	SIGNAL NAME	TEST POINT	P	COMMAND	CONDITION	COMMENTS
				ENABLE START S114	ESTAHL	
S114						
	ELCOLF (3.32)	M24-6	F	ENABLE LC RGTR	MOVE	
	CK1ALR (3.32)	N27-1	F	CLR K1 ADRS FF	MOVE	
	ELAOLF (3.32)			ENABLE LA RGTR	COMPARE	
	SK1ALR (3.32)	N27-2	F	SET K1 ADRS	COMPARE	
	CBKSCF (3.32)	N33-4	F	CLR BLOCK K ADRS FF		
				SET C1/C2 AOR FF	C12AOR	C12AOR = C ≥10
	SCMUCV (3.32)	O30-3	F	SET CMU ON FF		
	CBAOCX (3.32	L34-8	F	CLR BLOCK AOR FF		
				ENABLE C2≥C1 FF		
	ESCR (3.32)	M15-2	T	ENABLE SCR RGTR		
	ELESCF (3.32)			ENABLE LE RGTR		
	G462 (3.32)			SELECT L-ADD → I46	CMU ON FF	
	G411 (3.32)	N24-10	T	SELECT (C2 - C1) → C-ADDER		
	$\overline{G440}$. $\overline{G441}$ (3.30)			SELECT - 12_8 → I44		
	G401 (3.32)	N23-8	T	SELECT C1 → I40		
				ENABLE START SEQ - S164	$\overline{C12AOR}$	
S164						
	G400 (3.32)	N23-10	T	SELECT C2 → I40		
				SET C1/C2 AOR FF	C12AOR	
	ESCR (3.32)	M15-2	T	ENABLE SCR RGTR	C1 > C2	
	G410 (3.32)	N24-11	T	SELECT SCR + 12 → C-ADDER	MOVE . C1 > C2	
	$\overline{G410}$. $\overline{G411}$ (3.32)			SELECT (C1 - C2) → C-ADDER	COMPARE . C1 > C2	
				ENABLE START SEQ - 214	$\overline{C12AOR}$	

TABLE 5-2-21. COMPARE/MOVE COMMAND TIMING

SEQUENCE: START (cont.)

Page #2 of 2

TIME	SIGNAL NAME	TEST POINT	P	COMMAND	CONDITION	COMMENTS
S214						
	LA45LV (3.32)	M27-1	F	SELECT LA → I45	MOVE	
	G401 (3.32)	N23-8	T	SELECT C1 → I40	MOVE	
	$\overline{\text{LA45LV}}$ (3.32)			SELECT LC → I45	COMPARE	
	G400 (3.32)	N23-10	T	SELECT C2 → I40	COMPARE	
	ESRSCF (3.32)	N25-14	F	ENABLE CSR RGTR		
	ELESAF (3.32)			ENABLE LE RGTR		
	ELFSAF (3.32)	M24-13	F	ENABLE LF RGTR		
	OJJSCF (3.32)	N25-4	F	SELECT 0 → I40 DECODER		
	4044CF (3.32)			SELECT I40 → I44		
	F46X (3.9)			SELECT SCR → I19		
	SLI91 (3.9)			SELECT I19 → I9		
				ENABLE START SEQ - S264	S214	
S264						
	ELA0LF (3.32)			ENABLE LA RGTR	MOVE	
	ELC0LF (3.32)	M24-6	F	ENABLE LC RGTR	COMPARE	
	ELESBF (3.32)			ENABLE LE RGTR		
	EPWNCF (3.32)			ENABLE PW RGTR		
	EQ0SCF (3.328)	L25-7	F	ENABLE Q RGTR		
	BLKR (3.38)	M08-8	F	BLOCK R RGTR OUTPUT		
	$\overline{\text{G470}}$. G471 (3.30)			SELECT $\overline{\text{CSR}}$ → I47		
	F46X (3.9)			SELECT SCR → I19		
	SLI91			SELECT I19 → I9		
	ESC (3.9)			ENABLE SK RGTR		
	G400 . G401 (3.32)			SELECT SCR → I40		
	$\overline{\text{G440}}$. $\overline{\text{G441}}$ (3.30)			SELECT -12_8 → I44		
	G462 (3.31)			SELECT L-ADDER → I46		



CONTROL DATA
CANADIAN DEVELOPMENT DIVISION

COMPARE MOVE INSTRUCTION DECODE START SEQUENCE

CODE IDENT.		DWG. NO.	REV
34570	D	19981800	K
PUB. NO.		SHEET CPU 3.32	PAGE NO. 5-2-67

CARLETON INSTRUMENTS 138-3 200-13-74

DETAILED PAK DIAGRAM (CPU 3.33, 3.34, 3.35, 3.36, 3.37)

ADDRESS SEQUENCE

Initially, the address sequence is enabled by the start sequence clearing the block K address FF located on the JS module (CPU 2.34). The K1 address FF located on the JR module (CPU 3.34) is set during the start sequence by SK1ALR for a compare instruction, or reset during the start sequence by CK1ALR for a move instruction. Also, the 1st address FF located on the JS module will be set by CMU master clear (CMUCC) which is activated at the beginning and end of every CMU instruction.

MOVE INSTRUCTION (464, 465 - Refer to timing diagram, figure 5-2-32)

1st ADDRESS

1. The contents of the K2 register are loaded in the F register of the small adder. RA is loaded into E. The resultant relative address from the small adder is stored in the F register and transmitted to central memory control.

The transmission of a K address and memory request to central memory control is enabled by address sequence K264. The K264 timing chain is enabled only by specific conditions to ensure a valid address is being transmitted. 1st address FF enables K264 and, assuming the address transmitter register is not full (indicated by ADRS XMIT FF reset), the enable address transmit signal (EATOR) will be generated.

2. During 1st address, the length in LC will be decremented by -12_8 in the L adder. This is performed to test for a K2 exhaust on the first word. If K2 has exhausted (that is, LC $\leq$ 12), the K2 exhaust FF located on the JQ module (CPU 3.36) will be set, and the 1st & last FF will be set. The 1st & last FF will enable the short data sequence. On the 1st address, the decremented length count will not be transferred to the LC register. LC will remain at its original value.

3. The data counter located on the HT module (CPU 3.39) is incremented by one. The increment is enabled by the update data counter signal (UPDKPJ) from the JP module (CPU 3.33). The counter contains a count representing the number of words requested from memory. As each word is received, the count is decremented by one.

4. Clear 1st address FF, set 2nd address FF, clear K1 address FF (CPU 3.35).

2nd ADDRESS

1. With the K1 address FF set, K1 will be addressed in a manner similar to step 1 above.

The K264 address sequence will not be enabled until central memory control has generated an accept for the previous memory request. The accept will reset the ADRS XMIT FULL FF (ATFNSR) located on the JS module (CPU 3.35).

2. The length in LA will be decremented by -12_8 in the L adder. The group carry bit (LADDG) from the L adder carry look-ahead network is monitored on the JQ module (CPU 3.36). Absence of a carry indicates an exhaust condition and will set the K1 exhaust FF. If LA>12_8, the enable LA signal ELA0QF will allow the decremented count to be stored back in the LA register.

3. Increment data counter - described in step 3 above.

4. The buffer counter located on the JV module (CPU 3.37) is incremented by one. The increment is enabled by the update buffer counter signal (UPBKPV) from the JP module (CPU 3.36). The counter contains a count representing the number of K1 words requested from memory. For every word written into K2, the buffer counter will be decremented by one.

5. Clear 2nd address FF, set 1st write FF located on JS module (CPU 3.35).

The K1 address FF will determine whether the address sequence is to perform additional K1 read requests, or formulate the first K2 write address. The K1 address FF, located on the JR module (CPU 3.34), will remain set until buffer counter reaches a count of 5, or the K1 address has been exhausted. At that point, the K1 address FF is reset and, with the 1st write FF set, the address sequence will prepare the K2 address.

1st WRITE. K1 ADRS. $\overline{\text{1st ADRS}}$. $\overline{\text{2nd ADRS}}$

1. With the K1 address FF set, the contents of the K1 register are gated to the E register, +1 is forced to the F register by the 1TOFN signal generated from the JP module (CPU 3.33). The result K1 + 1 is returned to the K1 register by the enable K1 signal EK1NRC. RA is added to the contents of F and the resultant K1 address is transmitted to central memory control.

The remainder of the sequence will be the same as steps 2, 3 and 4 of 2nd address described above.

1st WRITE. $\overline{\text{K1 ADRS}}$. $\overline{\text{1st ADRS}}$. $\overline{\text{2nd ADRS}}$

1. With the K1 address FF reset, K2 is gated to F, RA is gated to E. The result from the small adder produces a relative memory address for the first K2 word.

The tramsmission of the K2 address and a memory write request will not occur until the HR register is loaded with a data word to be written into memory. Loading of the HR register is controlled by the data sequence. With the HR register loaded, the HR full FF is set to enable K264 of the address sequence. K264, in turn, enables address transmit (EATOR) and data transmit by generating EDTOS from the JS module (CPU 3.35).

2. The length in LC will be decremented by -12_8 in the L adder. This is performed to test for K2 exhaust. If LC $\geq 12_8$, the enable LC signal ELCOQF generated from the JQ module will allow the decremented count to be stored back in the LC register.

3. The sequence then performs a double subtraction that monitors whether the next K2 sequence will exhaust the length in LC. The look-ahead function allows the address sequence to read the last K2 word before performing the last K2 write. The decremented contents of LC from the L adder are enabled to the LF register via I45, and thus the second subtraction is performed. The absence of a group carry or pass ($\overline{\text{LADDG}}$, $\overline{\text{LADDP}}$) from the L adder indicates L < 12; if K1 has already been exhausted, the LAC < 12 FF will be set.

Setting of the LAC < 12 FF enables setting the K1 address FF. The address sequence will thus perform a K1 read sequence on the next cycle to obtain the last word of K2. Since K1 must have exhausted in order to set LAC < 12, the current K2 address used to produce the memory address in step 1 is stored in K1 by the EKINRC signal on the JR module. Thus both K1 and K2 will contain the same K2 address.

4. Reset 1st write FF

If the K1 exhaust FF is not set, the address sequence will toggle between K1 read and K2 write until K1 has exhausted. The address sequences for K1 and K2 are identical to those already described but with two exceptions, (a) and (b), listed below.

$\overline{\text{1st ADRS}}$. $\overline{\text{2nd ADRS}}$. K1 ADRS

(a) K1 is gated to E, +1 is forced to F, result K1 + 1 returned to K1.
(b) Data counter and buffer counter are incremented by one.

Once K1 has exhaused, the address sequence will perform K2 write until the LAC < 12 FF is set, at which time the last K2 read is performed.

LAC < 12 FF. K1 ADRS (Last K2 READ)

1. With the K1 address FF set, the contents of the K1 register (K1 will contain the previous K2 address) are gated to the E register, +1 is forced to the F register. The result from the small adder is added to RA to produce a relative memory address for the last K2 word. This address is transmitted to central memory control.

2. The data counter is incremented by one. The increment is enabled by the update data counter signal (UPDKQJ) from the JQ module (CPU 3.36).

3. Reset K1 address FF.

With the last K2 read performed, the address sequence will generate the K2 address for the last K2 write.

LAC < 12. $\overline{\text{K1 ADRS}}$ (Last K2 WRITE)

1. With the K1 address FF reset, the contents of the K2 register are gated to the E register, +1 is forced to the F register. The result from the small adder is added to RA to produce a relative memory address for the last K2 write.

The transmission of the last K2 address and a memory write request will not occur until the HR full FF is set by the data sequence.

COMPARE INSTRUCTION (466, 467 - Refer to timing diagram, figure 5-2-32)

The address sequencing for a compare instruction is similar to that of a move with the following exceptions:

1st ADDRESS

1. K1 is addressed rather than K2, as in a move.
2. LA is decremented by -12_8. K1 exhaust test is performed. The 1st & last FF cannot be set during 1st address if K1 has exhausted. If C1 > C2, the LA value before it is decremented is stored in LAC1 and the previous contents of LAC1 are stored in LAC2.
3. The data counter and buffer counter are incremented by one.

2nd ADDRESS

1. K2 is addressed rather than K1, as in a move.
2. LC is decremented by -12_8. K2 exhaust test is performed. If K2 has exhausted and K1 exhausted on the previous sequence, the 1st & last FF is set to enable the short data sequence. If C2 ≥ C1, the LC value before it is decremented is stored in LAC1, and the previous contents of LAC1 are stored in LAC2.
3. The data counter and buffer counter are incremented by one.

With the buffer counter equal to 4, the block K ADRS FF (CPU 3.35) is set. The block K ADRS FF will prevent further address sequences from occurring until the compare sequence has compared the first pair of words. The compare sequence decrements the buffer counter by two and resets the block K ADRS FF.

Address sequencing will continue until K1 and K2 exhaust, or a compare unequal occurs.

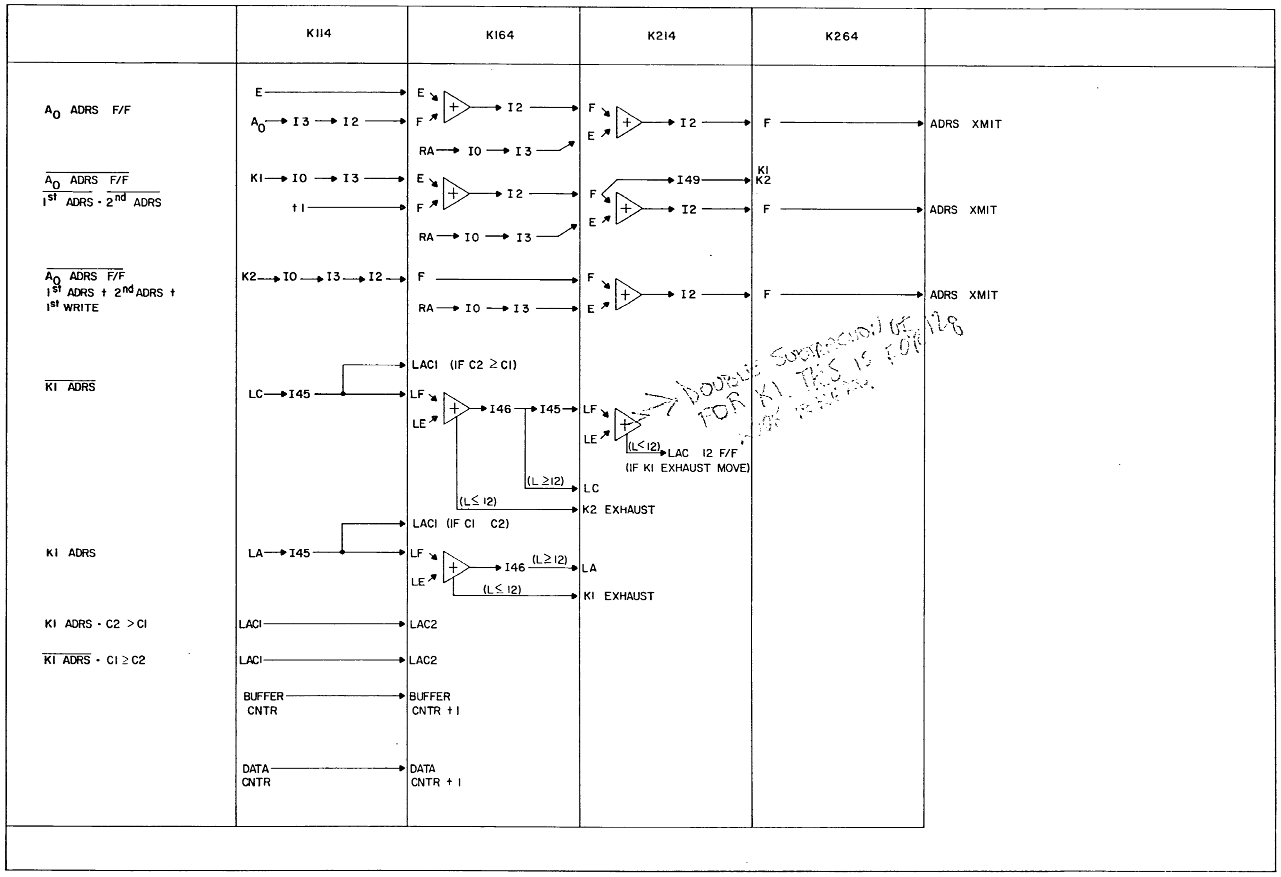
COMPARE COLLATE - COLLATE TABLE LOOK-UP - A0 ADRS

The collate sequence will set the A0 ADRS FF and clear the block K ADRS FF, allowing the address sequence to generate the correct table word address.

1. The collate sequence will load the upper bits (3-5) of the TQ or TS register in the E register. The address sequence will load the contents of the A0 address register in the F register. The result from the small adder is added to RA to produce the relative memory address for the desired table word.

2. The address sequence will clear the A0 ADRS FF and set the block K ADRS FF.

Further address sequencing will be blocked, unless the collate sequence sets the A0 ADRS FF and clears block K ADRS once again.



CONTROL DATA
CANADIAN DEVELOPMENT DIVISION

ADDRESS SEQ

CODE IDENT		DWG NO	REV
34570	D	19981800	A
FIGURE 5-2-32		PAGE NO 5-2-68-3	

CARLETON INSTRUMENTS 138 2 200-1274

TABLE 5-2-22. COMPARE/MOVE COMMAND TIMING

SEQUENCE: ADDRESS

Page 1 of 2

TIME	SIGNAL NAME	TEST POINT	P	COMMAND	CONDITION	COMMENTS
K100/ 114	(3.33)			ENABLE K100/K114	$\overline{\text{K BUSY FF}}$. $\overline{\text{F FULL FF}}$. $\overline{\text{BLOCK K ADRS FF}}$. $\overline{\text{A0RI}}$	
	(3.33)			SET K BUSY FF		
	ELFKPF (3.33)			ENABLE LF RGTR		
	AI3N (3.33)			SELECT A → I3	A0 ADRS FF	
	I3I2N (3.33)			SELECT I3 → I2	A0 ADRS FF + 1st ADRS + 2nd ADRS + 1st WRITE. $\overline{\text{K1 ADRS}}$	
	1TOFN (3.33)			SET +1 → F RGTR	$\overline{\text{A0 ADRS FF}}$. $\overline{\text{1st ADRS}}$. $\overline{\text{2nd ADRS}}$. (K1 ADRS + $\overline{\text{1st WRITE}}$)	$\overline{\text{I3 → I2}}$
	I0I3N (3.33)			SELECT I0 → I3	$\overline{\text{A0 ADRS FF}}$	
	EEK114 (3.33)			ENABLE E RGTR	$\overline{\text{A0 ADRS FF}}$	
	K1I0 QC (3.33)			K1 → I0	$\overline{\text{A0 ADRS FF}}$	
	NLA45V (3.33)	M27-5	F	SELECT LA → I45	$\overline{\text{A0 ADRS FF}}$. K1 ADRS FF	
	K2I0PC (3.33)			SELECT K2 → I0	$\overline{\text{A0 ADRS FF}}$. $\overline{\text{K1 ADRS FF}}$	
	$\overline{\text{LA45CV}}$ (3.33)	M27-5	T	SELECT LC → I45	$\overline{\text{A0 ADRS FF}}$. $\overline{\text{K1 ADRS FF}}$	
	EL1KPC (3.33)	M33-9	T	ENABLE LAC1 RGTR	$\overline{\text{A0 ADRS FF}}$. (C2 ≥ C1. $\overline{\text{K1 ADRS}}$ + C1 ≥ C2. K1 ADRS FF)	
	EL1KPC (3.33)	M33-9	T	ENABLE LAC2 RGTR	$\overline{\text{A0 ADRS FF}}$. (C2 ≥ C1. $\overline{\text{K1 ADRS}}$ + C1 ≥ C2. K1 ADRS FF)	
	UPDKPJ (3.33)	M32-7	F	INCREMENT DATA COUNTER	$\overline{\text{A0 ADRS FF}}$. [COMPARE + 1st ADRS + (K1 ADRS. $\overline{\text{LAC<12FF}}$)]	
	UPBKPV (3.33)	M27-8	T	INCREMENT BUFFER COUNTER	$\overline{\text{A0 ADRS FF}}$. [COMPARE + K1 ADRS FF. 2nd ADRS FF. C2 ≥ C1 + $\overline{\text{2nd ADRS}}$. $\overline{\text{LAC<12 FF}}$)]	
	(3.33)			ENABLE K164	K114. $\overline{\text{A0RI}}$	
K164				SELECT RA → I0		
	I0I3N (3.33)			SELECT I0 → I3		
	G462 (3.31)			SELECT L ADDER → I46		
	EEK114 (3.33)			ENABLE E RGTR		
	EL2KQF (3.36)	M24-2	F	ENABLE LAC2	2nd ADRS	
	EFN (3.36)			ENABLE F RGTR	$\overline{\text{1st ADRS}}$. $\overline{\text{2nd ADRS}}$. (K1 ADRS + $\overline{\text{1st WRITE}}$)	
	4645QU (3.36)			SELECT I46 → I45	$\overline{\text{1st ADRS}}$	
	ELFKQF (3.36)	L25-12	F	ENABLE LF RGTR	$\overline{\text{1st ADRS}}$	
	ELA0QF (3.36)	M24-5	F	ENABLE LA RGTR	$\overline{\text{A0 ADRS FF}}$. K1 ADRS. $\overline{\text{LAC<12 FF}}$. $\overline{\text{L<12}}$	
	ELC0QF (3.36)			ENABLE LC RGTR	$\overline{\text{A0 ADRS FF}}$. $\overline{\text{K1 ADRS}}$. (COMPARE + $\overline{\text{1st ADRS}}$) . $\overline{\text{L<12}}$	
	(3.36)			ENABLE K1 EXHAUST FF	$\overline{\text{A0 ADRS}}$. K1 ADRS. $\overline{\text{LAC<12}}$	
	(3.36)			ENABLE K2 EXHAUST FF	$\overline{\text{A0 ADRS}}$. $\overline{\text{K1 ADRS}}$	INPUT is L<12

TABLE 5-2-22. COMPARE/MOVE COMMAND TIMING

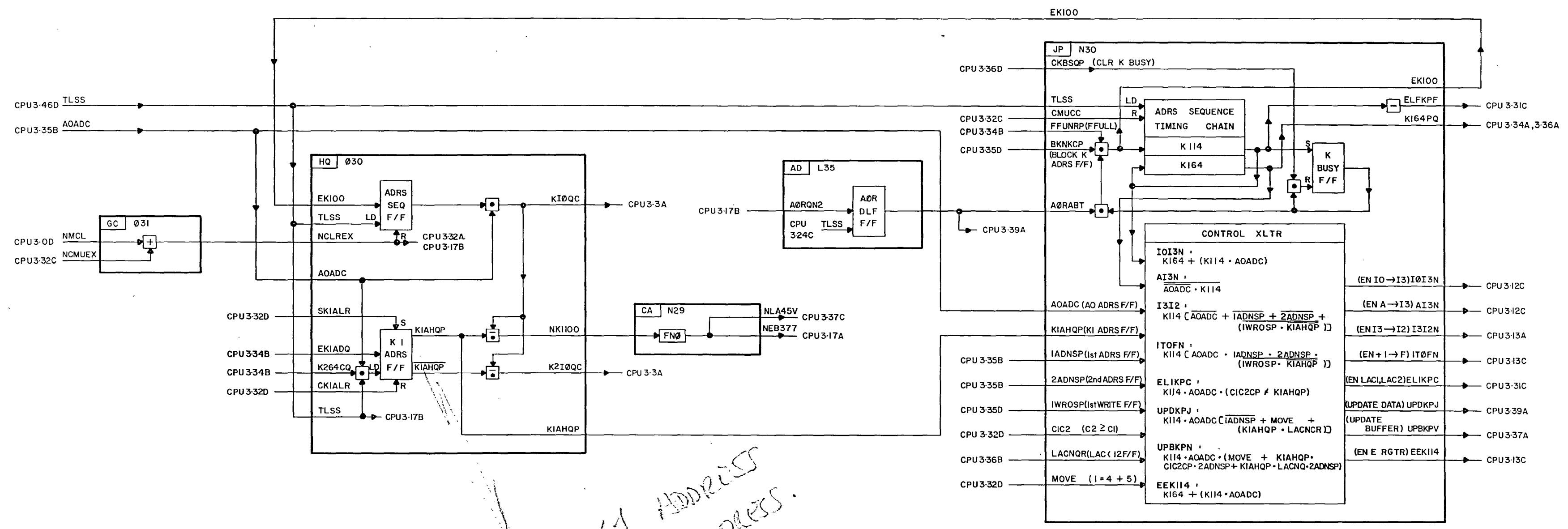
SEQUENCE: ADDRESS (cont.)

Page 2 of 2

TIME	SIGNAL NAME	TEST POINT	P	COMMAND	CONDITION	COMMENTS
	(3.34)			ENABLE K214	K164.$\overline{\text{AORI}}$	
K214						
	EFN (3.36)			ENABLE F RGTR		
	(3.34)			SET F FULL FF		
	CKBSQP (3.36)	N30-4	F	CLR K BUSY FF		
	G492 (3.3, 3.4)			SELECT F → I49		
	EK1NRC (3.34)			ENABLE K1 RGTR	$\overline{\text{A0 ADRS FF}}$.$\overline{\text{1st ADRS FF}}$.$\overset{+}{\overline{\text{2nd ADRS FF}}}$. (K1 ADRS + K1 EXHAUST)	
	KK2NRC (3.34)			ENABLE K2 RGTR	$\overline{\text{A0 ADRS FF}}$.$\overline{\text{1st ADRS FF}}$.$\overline{\text{2nd ADRS FF}}$. $\overline{\text{K1 ADRS}}$.$\overline{\text{1st WRITE}}$	
	(3.36)			ENABLE 1st & LAST FF	MOVE.1st ADRS + COMPARE.2nd ADRS	*NOTE INPUT AT BOTTOM
	(3.36)			ENABLE LAC<12 FF	MOVE.$\overline{\text{K1 ADRS}}$.$\overline{\text{A0 ADRS}}$.K1 EXHAUST. $\overline{\text{K2 EXHAUST}}$	INPUT IS L<12
	UPDKQJ (3.36)	M32-3	F	INCREMENT DATA COUNTER	MOVE.K1 ADRS.$\overline{\text{A0ADRS}}$.K1 EXHAUST. LAC<12 FF. L<12.$\overline{\text{1st ADRS}}$	
	(3.34)			ENABLE K264	F FULL.[$\overline{\text{ADRS XMIT FULL}}$.K1 ADRS + COMPARE + (HR FULL.$\overline{\text{K1 ADRS}}$) + 1st ADRS].$\overline{\text{A0RI}}$	
K264	(3.35)			SET ADRS XMIT FULL FF		
	(3.34)			CLR F FULL FF		
	EATOR (3.34)			ENABLE ADRS XMIT		MEMORY REQUEST
	EDTOS (3.35)			ENABLE DATA XMIT	MOVE.$\overline{\text{K1 ADRS}}$.$\overline{\text{1st ADRS}}$	WRITE BIT
	(3.3, 3.4)			SET ENABLE EXIT FF	K1.K2 EXHAUST	
	(3.34)			SET K1 ADRS XMIT FF	MOVE.1st ADRS	
	(3.34)			ENABLE K1 ADRS XMIT FF	$\overline{\text{A0 ADRS}}$.($\overline{\text{1st ADRS}}$ + COMPARE)	
	(3.35)			SET BLOCK K ADRS FF	K1.K2 EXHAUST + COMPARE.BUF4VS	BUF4VS = BUF CNTR = 4
	(3.35)			CLR 1st ADRS FF	1st ADRS	
	(3.35)			SET 2nd ADRS FF	1st ADRS	
	(3.35)			CLR 2nd ADRS FF	2nd ADRS	
	(3.35)			SET 1st WRITE FF	MOVE.2nd ADRS	
	(3.35)			CLR 1st WRITE FF	1st WRITE.$\overline{\text{K1 ADRS}}$	
				ENABLE K1 ADRS FF	$\overline{\text{A0 ADRS}}$	*NOTE INPUT AT BOTTOM
	(3.35)			CLR A0 ADRS FF	A0 ADRS	
	(3.35)			CLR A_2 FF	A0 FF	

INPUT TO 1st & LAST FF - MOVE.K2 EXHAUST + COMPARE.(K1.K2 EXHAUST)

INPUT TO K1 ADRS FF - [COMPARE.($\overline{\text{K1 EXHAUST}}$.$\overline{\text{K1 ADRS}}$ + K2 EXHAUST)] + [MOVE.($\overline{\text{K1 EXHAUST}}$.$\overline{\text{K1 ADRS}}$ + LAC<12 FF.$\overline{\text{K1 ADRS}}$ + $\overline{\text{K1 EXHAUST}}$.$\overline{\text{BF=5}}$.$\overline{\text{LAC<12}}$]



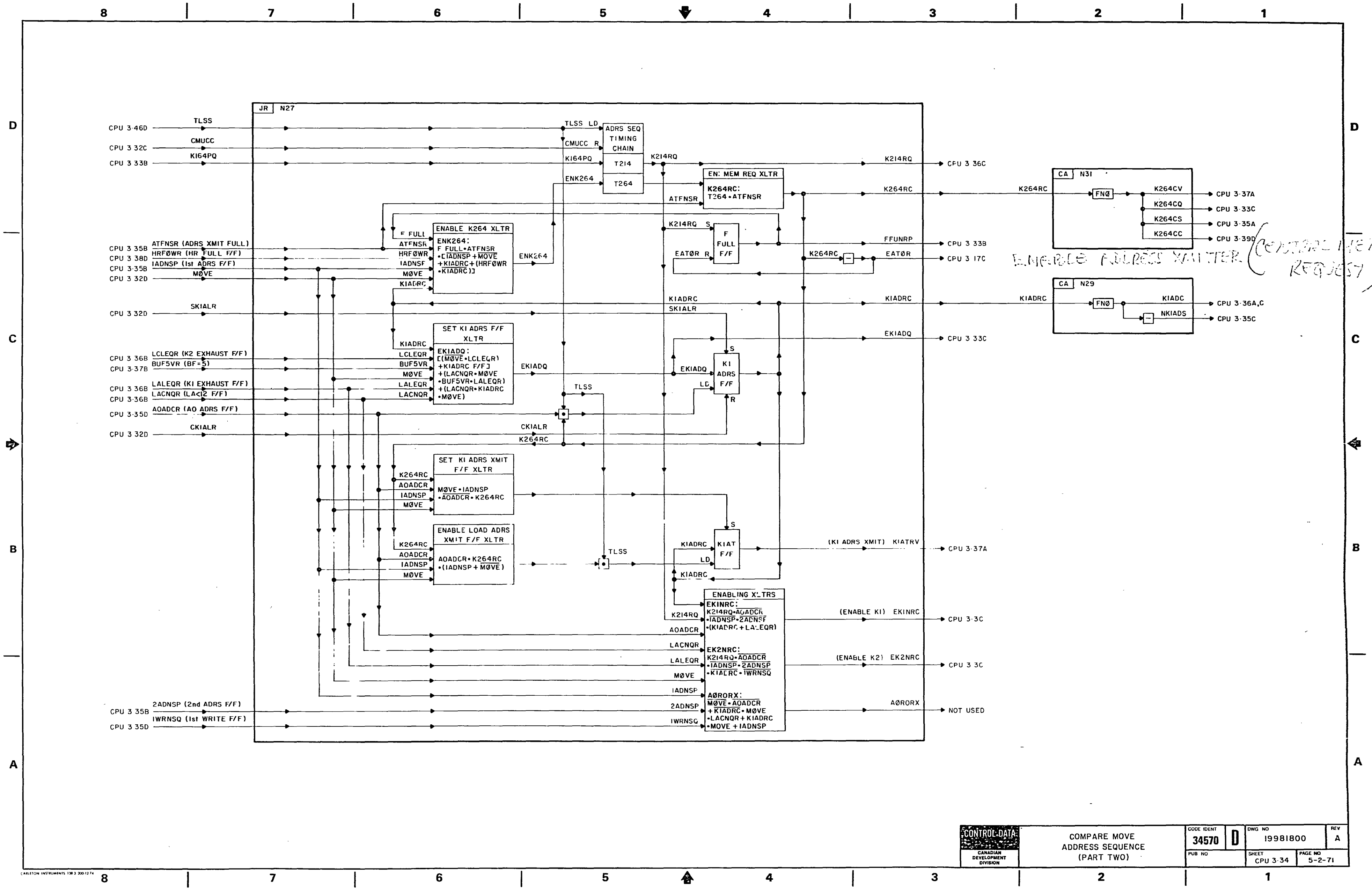
SET = K1 ADDRESS
CLR = K2 ADDRESS.

CONTROL DATA
CANADIAN DEVELOPMENT DIVISION

COMPARE MOVE
ADDRESS SEQUENCE
(PART ONE)

CODE IDENT.		DWG. NO.	REV
34570	D	19981800	K
PUB. NO.		SHEET CPU 3-33	PAGE NO. 5-2-69

CARLETON INSTRUMENTS 138-3 300-12-74



8 7 6 5 4 3 2 1
D C B A
JR N27
CPU 3-46D TLSS
CPU 3-32C CMUCC
CPU 3-33B K164PQ
TLSS LD
CMUCC R
K164PQ
ENK264
ADRS SEQ TIMING CHAIN
T214
T264
K214RQ
EN: MEM REQ XLTR
K264RC: T264•ATFNSR
ATFNSR
K214RQ CPU 3-36C
K264RC
CA N31
FNØ
K264CV CPU 3-37A
K264CQ CPU 3-33C
K264CS CPU 3-35A
K264CC CPU 3-39D
CPU 3-35B ATFNSR (ADRS XMIT FULL)
CPU 3-38D HRFØWR (HR FULL F/F)
CPU 3-35B IADNSP (1st ADRS F/F)
CPU 3-32D MØVE
F FULL
ATFNSR
HRFØWR
IADNSP
MØVE
KIADRC
ENABLE K264 XLTR
ENK264: F FULL•ATFNSR •[IADNSP+MØVE +KIADRC+(HRFØWR •KIADRC)]
ENK264
K214RQ S
F FULL F/F
EATØR R
FFUNRP CPU 3-33B
K264RC
EATØR CPU 3-17C
CA N29
KIADRC
FNØ
KIADC CPU 3-36A,C
NKIADS CPU 3-35C
CPU 3-32D SKIALR
KIADRC
SKIALR
KIADRC
EKIADQ CPU 3-33C
CPU 3-36B LCLEQR (K2 EXHAUST F/F)
CPU 3-37B BUF5VR (BF=5)
CPU 3-36B LALEQR (K1 EXHAUST F/F)
CPU 3-36B LACNQR (LAC2 F/F)
CPU 3-35D AOADCR (AO ADRS F/F)
CPU 3-32D CKIALR
SET KI ADRS F/F XLTR
KIADRC
LCLEQR
BUF5VR
MØVE
LALEQR
LACNQR
EKIADQ: [(MØVE•LCLEQR) +KIADRC F/F] +(LACNQR•MØVE •BUF5VR•LALEQR) +(LACNQR•KIADRC •MØVE)
EKIADQ
TLSS
EKIADQ
LD
S
KI ADRS F/F
R
CKIALR
K264RC
SET KI ADRS XMIT F/F XLTR
K264RC
AOADCR
IADNSP
MØVE
MØVE•IADNSP •AOADCR•K264RC
ENABLE LOAD ADRS XMIT F/F XLTR
K264RC
AOADCR
IADNSP
MØVE
AOADCR•K264RC •(IADNSP+MØVE)
TLSS
S
KIADRC
KIAT F/F
LD
KIADRC
(KI ADRS XMIT) KIATRV CPU 3-37A
ENABLING XLTRS
EKINRC: K214RQ•AOADCR •IADNSP•2ADNSP •(KIADRC+LALEQR)
K214RQ
AOADCR
LACNQR
LALEQR
MØVE
IADNSP
2ADNSP
IWRNSQ
EK2NRC: K214RQ•AOADCR •IADNSP•2ADNSP •KIADRC•IWRNSQ
AØRORX: MØVE•AOADCR +KIADRC•MØVE •LACNQR+KIADRC •MOVE+IADNSP
(ENABLE KI) EKINRC CPU 3-3C
(ENABLE K2) EK2NRC CPU 3-3C
AØRORX NOT USED
CPU 3-35B 2ADNSP (2nd ADRS F/F)
CPU 3-35D IWRNSQ (1st WRITE F/F)
ENABLE ADDRESS XMITTER (CENTRAL MEM REQUEST)

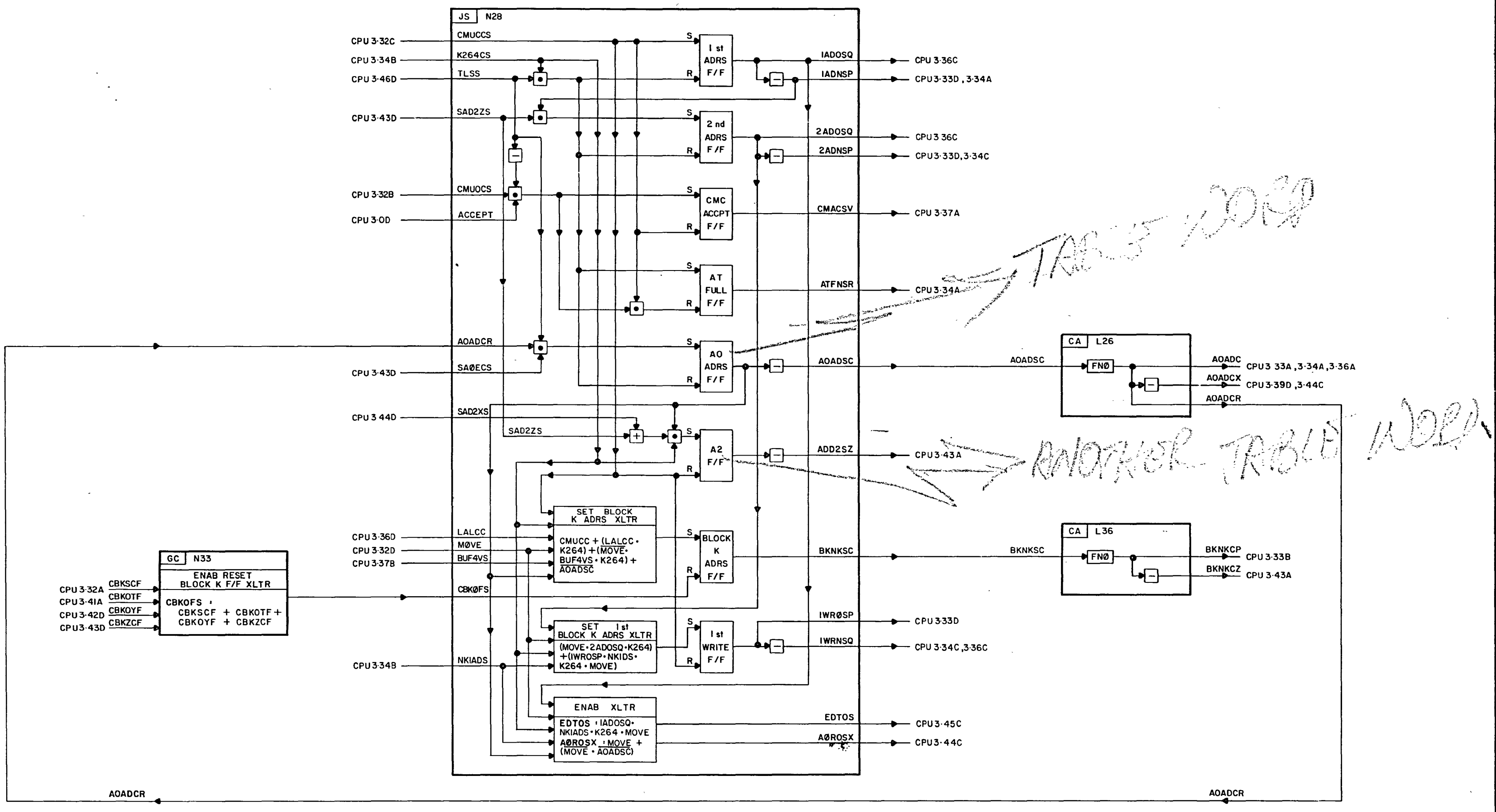
CONTROL DATA
CANADIAN DEVELOPMENT DIVISION

COMPARE MOVE
ADDRESS SEQUENCE
(PART TWO)

CODE IDENT		DWG NO	REV
34570	D	19981800	A
PUB NO		SHEET CPU 3-34	PAGE NO 5-2-71

CARLETON INSTRUMENTS 138 3 200 12 74

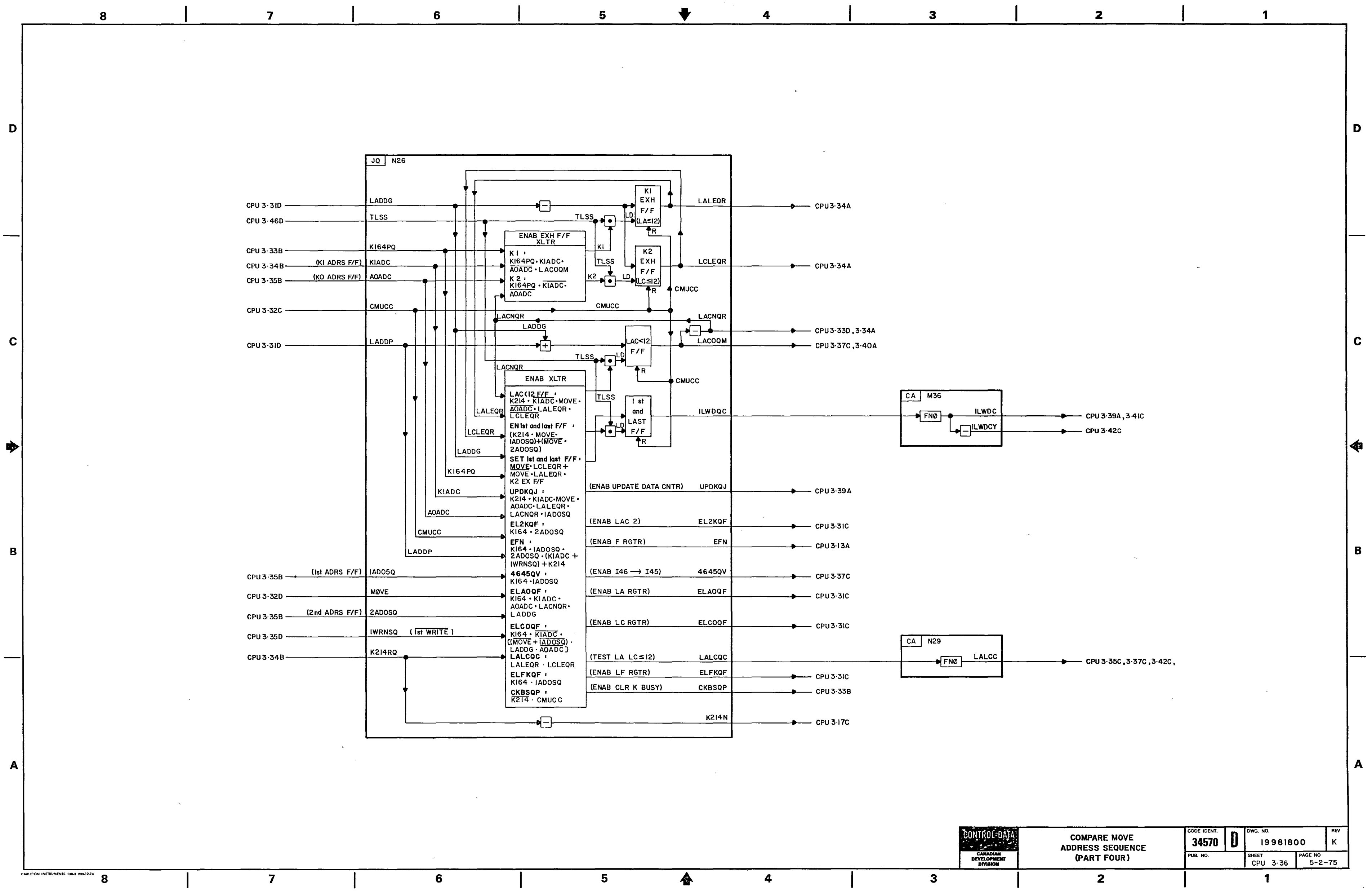
Central Memory Request Lxator



CONTROL DATA
CANADIAN DEVELOPMENT DIVISION

COMPARE MOVE
ADDRESS SEQUENCE
(PART THREE)

CODE IDENT		DWG NO	REV
34570	D	19981800	L
PUB NO		SHEET CPU 3·35	PAGE NO 5-2-73

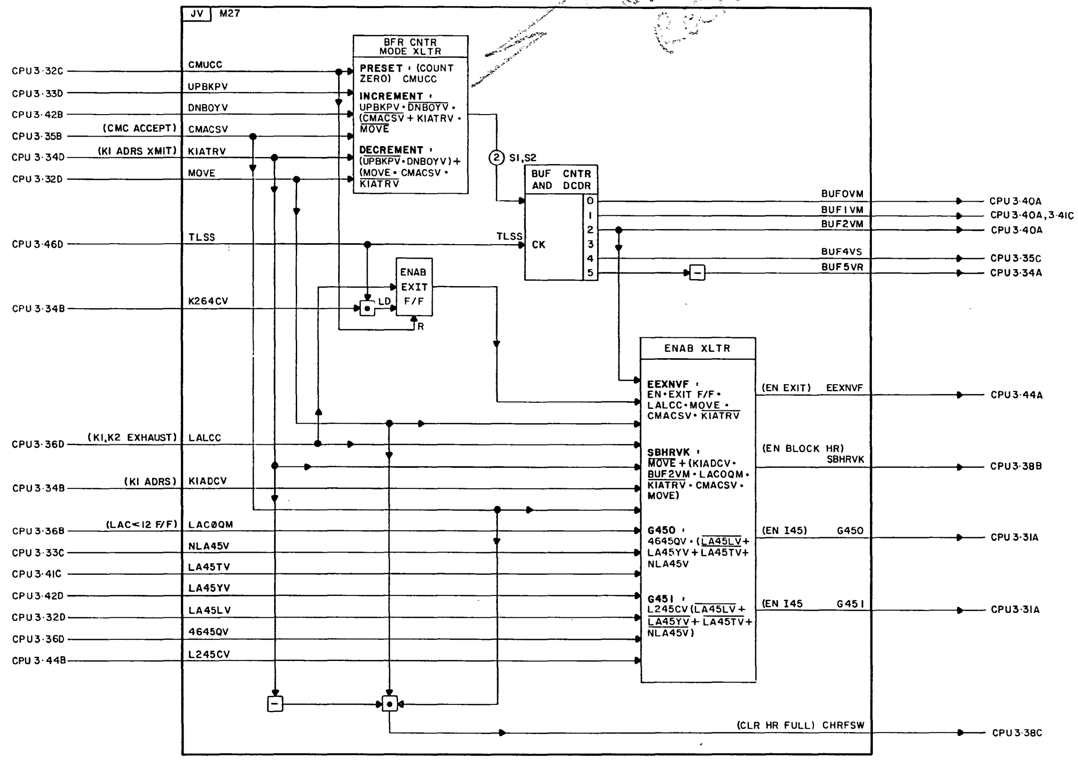


CONTROL DATA
CANADIAN DEVELOPMENT DIVISION

COMPARE MOVE
ADDRESS SEQUENCE
(PART FOUR)

CODE IDENT.		DWG. NO.	REV
34570	D	19981800	K
PUB. NO.		SHEET CPU 3-36	PAGE NO 5-2-75

CARLETON INSTRUMENTS 138-3 200-12-74



JV M27
BFR CNTR MODE XLTR
PRESET • (COUNT ZERO) CMUCC
INCREMENT • UPBKPV • DNBOYV • (CMACSV + KIATRV • MOVE
DECREMENT • (UPBKPV • DNBOYV) + (MOVE • CMACSV • KIATRV
BUF CNTR AND DCDR
CK
S1,S2
ENAB EXIT F/F
LD
R
ENAB XLTR
EEXNVF • EN • EXIT F/F • LALCC • MOVE • CMACSV • KIATRV
SBHRVK • MOVE + (KIADCV • BUF2VM • LACOQM • KIATRV • CMACSV • MOVE)
G450 • 4645QV • (LA45LV + LA45YV + LA45TV + NLA45V
G451 • L245CV (LA45LV + LA45YV + LA45TV + NLA45V)
CPU3·32C CMUCC
CPU3·33D UPBKPV
CPU3·42B DNBOYV
CPU3·35B (CMC ACCEPT) CMACSV
CPU 3·34D (KI ADRS XMIT) KIATRV
CPU3·32D MOVE
CPU3·46D TLSS
CPU 3·34B K264CV
CPU3·36D (KI,K2 EXHAUST) LALCC
CPU3·34B (KI ADRS) KIADCV
CPU3·36B (LAC<12 F/F) LACØQM
CPU3·33C NLA45V
CPU3·41C LA45TV
CPU3·42D LA45YV
CPU3·32D LA45LV
CPU3·36D 4645QV
CPU 3·44B L245CV
BUFOVM CPU3·40A
BUF1VM CPU3·40A,3·41C
BUF2VM CPU3·40A
BUF4VS CPU3·35C
BUF5VR CPU3·34A
(EN EXIT) EEXNVF CPU3·44A
(EN BLOCK HR) SBHRVK CPU3·38B
(EN I45) G450 CPU3·31A
(EN I45 G451 CPU3·31A
(CLR HR FULL) CHRFSW CPU3·38C
KEEPS TRACK OF NUMBER OF MEM. REQUESTS OF WORDS RECEIVED FROM MEMORY.

CONTROL DATA CANADIAN DEVELOPMENT DIVISION

COMPARE MOVE ADDRESS SEQUENCE (PART FIVE)

CODE IDENT.		DWG. NO.	REV
34570	D	19981800	A
PUB. NO.		SHEET CPU 3·37	PAGE NO. 5-2-77

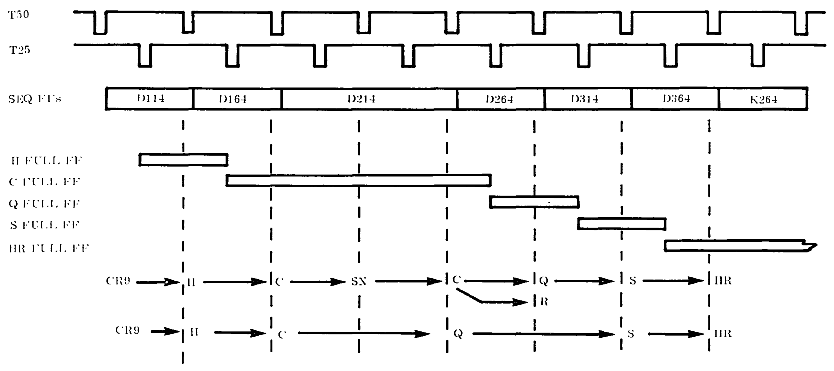
CARLETON INSTRUMENTS 138-3 200-12 74

DETAILED PAK DIAGRAM (CPU 3.38, 3.39, 3.40)

DATA SEQUENCE

Central memory control generates the data ready signal (DARDY), 50 ns before the transmission of requested data. The accept sequence located on the GM module (CPU 3.16) is conditioned by data ready. Data ready starts the accept sequence timing chain: DR50, DR64. The leading edge of DR50 generates central memory data ready (CMDRM) to the data sequence HT module (CPU 3.39). At the next clock, CMDRJX starts the data sequence timing chain: D164, D214, D264, D314, and D364.

The basic data path flow through the data sequence is shown on the illustration below.



The time interval between the leading edge of CMDRJX and the beginning of D164 is considered as D114.

During time intervals D114 through D364, data in CR9 can be propagated in succession through five registers (H, C, Q, S, and HR) with the loading of each controlled independently by the data sequence. The full conditions of these registers are monitored by five control FFs: H full, C full, Q full, S full, and HR full, located on the HT and JW modules (CPU 3.39; 3.38). Each control FF is enabled by a special 25 ns clock that occurs half way between the regular clock. A full condition alerts the next sequence interval to enable continuance of data propagation.

Normally, D214 allows for the realignment of character positions in C. Realigment is performed by right shifting the desired number of characters through the shift network and returning the shifted results to the C register. The C full FF remains set while the shift is taking place. When a realignment of data in C is not required, D214 stores the unshifted data from C directly into Q. Depending on sequence conditions, Q full may be set at this point.

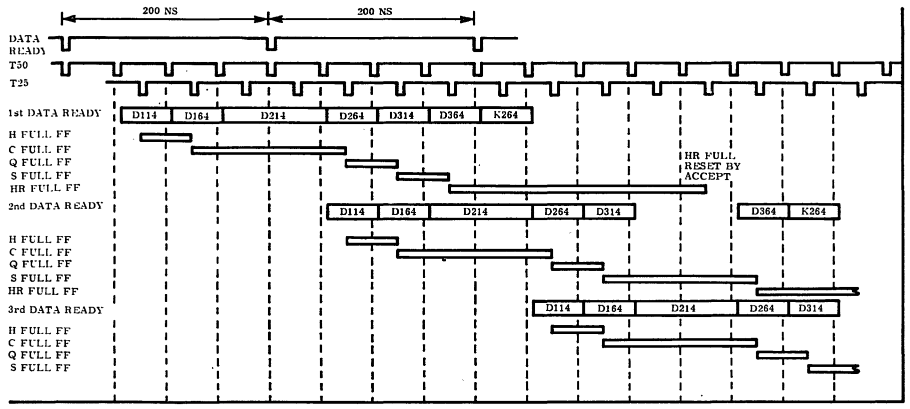
Normally, D264 allows for the storage of shifted data from C (bits 48-107) into Q. Any shift residue characters, (those shifted past the tenth character position, bit 48), are transferred into R for temporary storage.

Similarly, D314 allows for the storage of data from Q into S; D364 allows for the storage of data from S into HR. HR full alerts address sequence K264 to initiate a memory write request. Prior to the generation of HR full, the address sequence will already have placed a K2 address into F.

To prevent writing, a compare instruction blocks D 364 altogether.

Consecutive data transmission to central memory control is dependent on the receipt of a signal that acknowledges memory acceptance of the previous write data. Acceptance of a write request which is unduly delayed causes stacking of data in the S, Q, C and H registers. It is specifically because of this stacking capability that the address sequence monitors the buffer counter to ensure that only five K1 address requests ($C2 \geq C1$), or six K1 address requests ($C1 > C2$) can be issued prior to a write.

An example of data stacking is illustrated below.



Three data ready responses are received at their maximum rate of 200 ns. Each data ready initiates a data sequence.

The first data sequence proceeds through D364, where HR full is set. HR full enables address sequence K264, and it is during this time that first write request occurs. The second data sequence starts 200 ns after the first; it proceeds through D314 only. Since HR is full, D364 cannot be enabled. The second data word remains stacked in S with the S full FF set. The third data sequence starts 200 ns after the second; it proceeds through D214. By D214 time, HR full is reset by an accept for the first write request. At the next clock, D364 is enabled for the second word while D264 is enabled for the third. The second and third words are simultaneously transferred from S and C into HR and Q, respectively. HR full and Q full enable K264 and D314 at the next clock. K264 initiates a memory write request for the second word while D314 allows for the transfer of data from Q into S. The third data word will remain stacked in S until the second write accept is received.

MOVE INSTRUCTION (464, 465 - refer to Figure 5-2-33)

Three data detection FFs are utilized by the data sequence to determine path selection. The three FFs, 1st data, 2nd data and 3rd data are located on the HW module (CPU 3.40).

1st data sets at the beginning of a CMU instruction; it enables sequence path selection for receipt of the first word. 2nd data is set at the end of 1st data; it enables sequence path selection for receipt of the second word. Two paths are provided for 2nd data; the path chosen is dependent on the C2 ≥ C1 FF. 3rd data sets at the end of 2nd data when C1> C2; it enables sequence path selection for receipt of the third word.

1st DATA

Receipt of the first data ready response initiates the first data sequence. The first word received will be the first word of the K2 destination field.

1. Data ready allows the data counter (CPU 3.39) to be decremented by one.
2. The first K2 word is transferred into Q, where it will remain until second data. The entire Q register is enabled because CSR contained zero from the start sequence.
3. Clear 1st data, set 2nd data.

The data sequence is now conditioned for receipt of the second data word. The next data ready will initiate the second data sequence.

2nd DATA

The second word received will be the first word of the K1 source field. The path chosen for 2nd data is dependent on the C2 ≥ C1 FF. Each path is described separately.

C2 ≥ C1

With the C2 offset greater than, or equal to the C1 offset, the first K1 word is shifted to align the first actual character position of K1 with K2. The shifted K1 data from C is transferred into Q, where it combines with the K2 offset stored in Q from 1st data. The shift residue from C is transferred into R for temporary storage until the next sequence. The complete word in Q becomes the first K2 write data; it is transferred into S and HR, at which time the first write request is performed.

1. Data ready allows the data counter to be decremented by one (CPU 3.39).
2. The C2 offset in CSR controls loading Q, so that the K2 offset is protected while the shifted K1 occupies the remaining character positions of Q.
3. Clear 2nd data.

All subsequent data responses use the normal path ($\overline{\text{1st DATA}}$. $\overline{\text{2nd DATA}}$. $\overline{\text{3rd DATA}}$), until last word is detected.

C1 > C2

With the C1 offset greater than the C2 offset, the first K1 word is shifted to align the first actual character position of K1 with K2. Since a left shift cannot be performed, the shift will cause all characters to reside in the residue portion of C. The residue from C is transferred into R, where it will remain until the next sequence.

1. Data ready allows the data counter to be decremented by one (CPU 3.39).
2. Clear 2nd data, set 3rd data.

The data sequence is now conditioned for receipt of the third data word. The first K1 word that was aligned with K2 remains in the R register until the next K1 word is received.

3rd DATA

While the second K1 word is being shifted to align K1 with K2, the first residue is transferred from R into Q, where it combines with the K2 offset stored in Q from 1st data. The shifted second K1 word in C is then transferred into Q to occupy its remaining character positions. The shift residue from C is transferred into R for storage until the next sequence. The complete word in Q becomes the first K2 write data, and is transferred into S and HR, at which time the first write request is performed.

1. Data ready allows the data counter to be decremented by one (CPU 3.39).
2. The C2 offset in CSR controls the loading of Q, so that the K2 offset is protected while the first K1 residue from R is transferred into Q.
3. The shift count in PW controls the loading of Q, so that the K2 offset and K1 residue previously stored in Q are protected while the second K1 word occupies the remainder of Q.
4. Clear 3rd data.

All subsequent data ready responses use the normal path ($\overline{\text{1st DATA}}$. $\overline{\text{2nd DATA}}$. $\overline{\text{3rd DATA}}$, until last word is detected.

$\overline{\text{1st DATA}}$. $\overline{\text{2nd DATA}}$. $\overline{\text{3rd DATA}}$ (Normal Path)

The normal path operation is similar to 3rd data with the exception that the CSR register will contain zero instead of an offset value.

The data counter is decremented by one for each data ready. When the count equals zero and the LAC <12 FF is set, the data word in CR9 is the last K2 word. The data path chosen is dependent on the remaining buffer count value.

DT=0 . BF=1 . LAC < 12FF

A data count of zero and a buffer count of one indicate that the block HR FF must have been set on the previous sequence to prevent writing the last K2 word until the partial write characters from K2 are read. (Examples of this condition are illustrated in figures 5-2-34 and 5-2-36.)

The remaining length value in LC determines how many partial write characters from K2 must be returned to K2.

1. The remaining length value from LC is transferred into PW where it controls loading the partial write characters into Q.

DT=0 . BF=0 . LAC < 12 FF

A data count of zero and a buffer count of one indicate that the block HR FF was not previously set. Residue characters from the previous sequence stored in R are transferred into Q. The partial write characters from K2 are transferred into Q. (An example of this condition is illustrated in figure 5-2-35.)

1. The remaining length value from LC is transferred into PW and is used to control the loading of the partial write characters into Q.

COMPARE INSTRUCTION (466, 467 - Refer to figure 5-2-33)

A compare instruction sets the block HR FF (CPU 3.38); it remains set throughout the instruction to block D364.

The toggle FF located on the HW module (CPU 3.40) is used by compare to select the appropriate data path for K1 and K2. In its reset state, the toggle FF selects the K1 data path. K1 is always stored in S. When set, the toggle FF selects the K2 data path. K2 is always stored in Q.

The 1st data flip-flop is the only data detection FF utilized by compare. Path selections are determined by the condition of 1st data, C2 ≥ C1, toggle and last compare.

The first word received (of a pair) will always be a K1 word. The 1st data FF is set at the beginning of a CMU instruction; it enables sequence path selection for the first word received. Four paths are provided for 1st data: (C2 ≥ C1 . $\overline{\text{TOGGLE}}$), (C2 ≥ C1 . TOGGLE), (C1 > C2 . $\overline{\text{TOGGLE}}$), and (C1 > C2 . TOGGLE).

1st DATA . C2 ≥ C1 . $\overline{\text{TOGGLE}}$

The first K1 word received is shifted to align K1 with K2. The shift residue is transferred into the R register for storage until the next sequence. The shifted K1 data from C is transferred into Q and S.

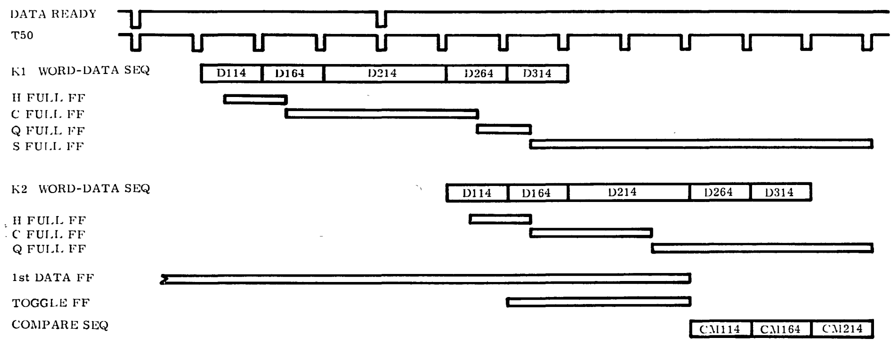
1. Data ready allows the data counter to be decremented by one (CPU 3.39).
2. The C2 offset in the CSR register prevents loading the C2 offset positions of Q.
3. The toggle FF is set; the 1st data FF remains set until receipt of the first K2 word.

1st DATA . C2 ≥ C1 . TOGGLE

The first K2 word received is transferred directly into Q without shifting, since K1 was already shifted to align with K2.

1. Data ready allows the data counter to be decremented by one (CPU 3.39).
2. The C2 offset in the CSR register prevents loading the C2 offset positions of Q.
3. Reset 1st data FF and toggle FF.
4. Enable compare sequence.

A representative timing diagram for 1st data is shown below.



All subsequent data ready responses will use the normal compare data path, ($\overline{\text{1st DATA}}$. $\overline{\text{LAST COMPARE}}$), until last compare is detected.

1st DATA . C1 > C2

With C1 > C2, the paths for 1st data are similar to those previously described for C2 ≥ C1, with the following exceptions:

1. Rather than K1 being shifted to align with K2, the opposite is performed; K2 is shifted to align with K1.

2. Because of this change, the loading of Q is controlled by C1 instead of C2.

All subsequent data ready responses will use the normal data path ($\overline{\text{1st DATA}}$. $\overline{\text{LAST COMPARE}}$), until last compare is detected.

$\overline{\text{1st DATA}}$. $\overline{\text{LAST COMPARE}}$ (Normal Path)

The normal paths for compare are similar to the 1st data paths previously described. However, the difference between 1st data and the normal path is that the residue in R from the previous sequence is transferred into Q while the shift is being performed for K1 (C2 ≥ C1) or K2 (C1 > C2). The shifted characters of K1 or K2 are then transferred into Q to combine with the residue, forming a complete word. The current shift residue is transferred into R for storage until the next sequence.

1. The CSR register will always contain zero so that the residue from R can be loaded into the entire Q register.

2. The PW register, which contains the shift count, ensures that only the shifted contents of K1 (C2 ≥ C1) or K2 (C1 > C2) are transferred into Q while the previous residue, (step 1), is protected.

The normal path is used for receipt of data until the last compare FF is set. Last compare is set during the compare sequence when K1 and K2 have been exhausted and the compare sequence determines that the second last pair of words are equal. Data path selection for last compare is determined by the condition of the C2 ≥ C1 FF, toggle FF and the remaining buffer count.

LAST COMPARE . BF=2

The buffer count of two with the last compare FF set, indicates that one remaining pair of words must be received and compared before the instruction is completed. (An example of this condition is illustrated in figure 5-2-37.)

The data sequence is similar to a normal path except that:

1. CSR contains the remaining length from LA ($C1 > C2$) or LC ($C2 \geq C1$); CSR prevents loading Q with characters that are not part of the K1 or K2 last word field.
2. CSR $\neq$ PW ensures that only the shifted K1 ($C2 \geq C1$) or K2 ($C1 > C2$) characters are transferred into Q while the previous residue is protected, and characters not part of the K1 or K2 field are blocked.

LAST COMPARE . BF=1

A buffer count of one with the last compare FF set, indicates that only one remaining word must be received. (An example of this condition is illustrated in figure 5-2-38.)

With $C2 \geq C1$, the remaining word will be from K2; whereas with $C1 > C2$, the remaining word will be from K1. CSR will contain the remaining length from LA ($C1 > C2$), or LC ($C2 \geq C1$); CSR prevents loading Q with characters that are not part of the K1 or K2 last word field.

COLLATE TABLE LOOK-UP

The data sequence is also used during the compare collate operation to store the collate table word into the T register. The appropriate collate character is selected via I34 and transferred to either the TS or TQ register.

CENTRAL MEMORY CONTROL - ACCEPT RESPONSE

Central memory control responds to a read or write request by generating an accept signal (CMACM) when the request is honored. CMACM sets the CMC accept FF (CPU 3.35) and clears the ADRS XMIT full FF (at full). The reset condition of at full allows initiation of another address sequence, unless the block K ADRS FF is set.

Buffer Counter

The CMC accept FF for a write operation $\overline{\text{(K1 ADRS FF)}}$ decrements the buffer counter by one. K1 ADRS XMIT FF, located on the JR module (CPU 3.34), prevents decrementing the buffer counter on a read accept. The buffer counter and decrement controls are located on the JV module (CPU 3.37).

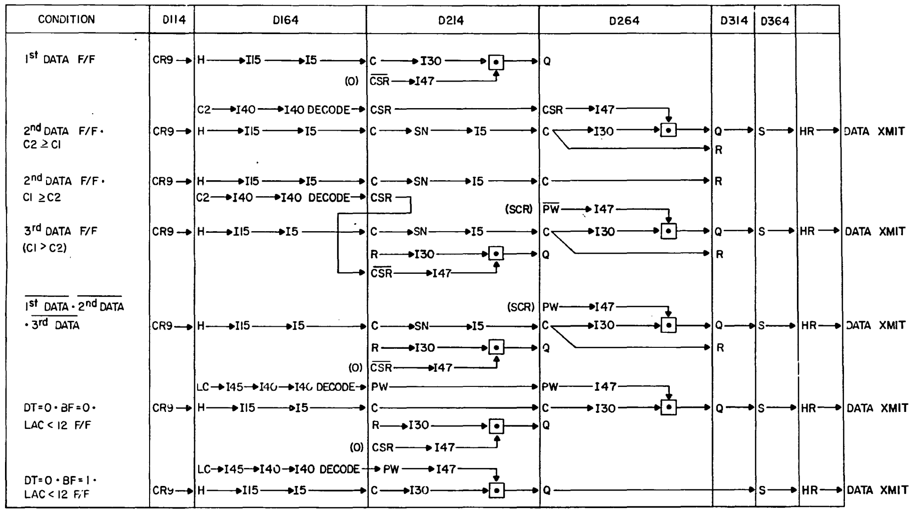
Block HR Controls

When the address sequence detects that the next K2 field length will exhaust (indicated by LAC < 12 FF) with a count of two in the buffer counter, the block HR FF will be set. LAC < 12 and a buffer count of two indicate that the last K1 word and last K2 word must be received before the last K2 write is performed.

The block HR FF is always set at the beginning of a compare instruction, and remains set throughout its execution.

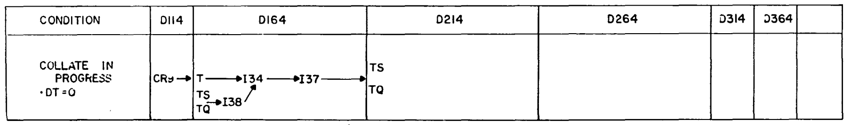
When set, the block HR FF blocks data sequence D364.

INSTRUCTION: MOVE (464 + 465)

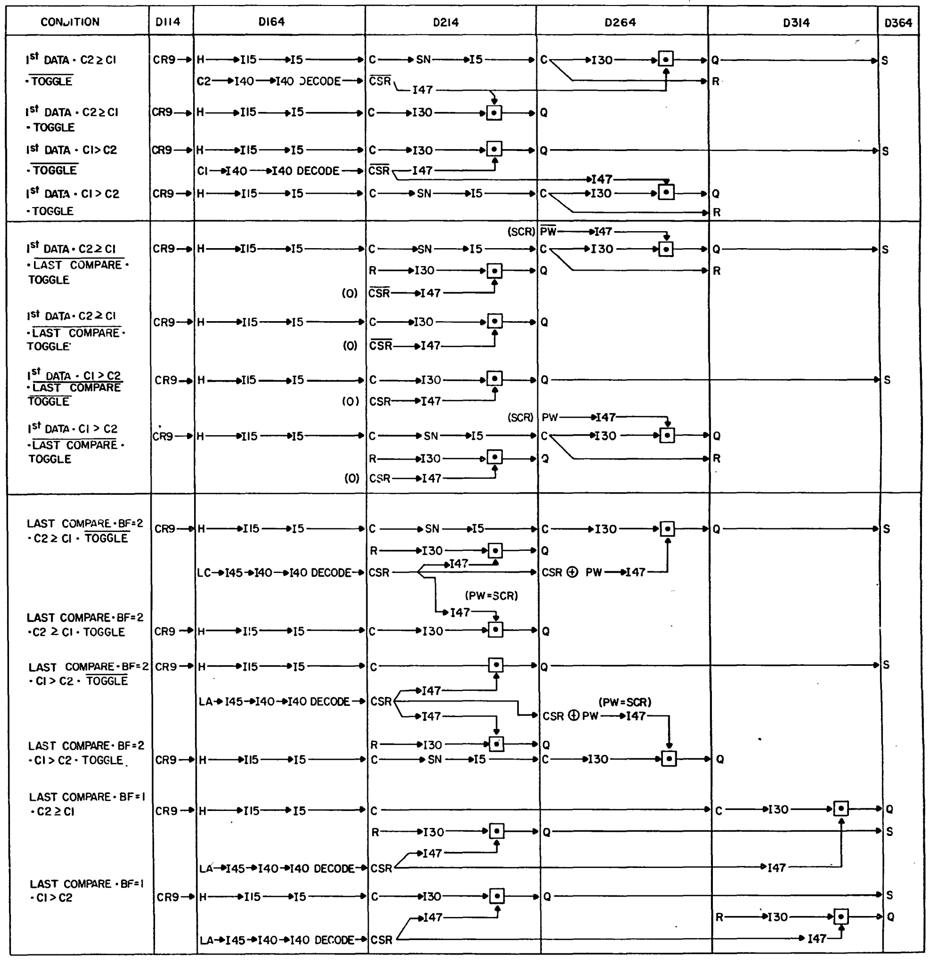


CONDITION
D114
D164
D214
D264
D314
D364
1st DATA F/F
CR9 → H → I15 → I5 → C → I30 → Q
(0) CSR → I47
2nd DATA F/F · C2 ≥ C1
C2 → I40 → I40 DECODE → CSR → CSR → I47
CR9 → H → I15 → I5 → C → SN → I5 → C → I30 → Q → S → HR → DATA XMIT
R
2nd DATA F/F · C1 ≥ C2
CR9 → H → I15 → I5 → C → SN → I5 → C → R
C2 → I40 → I40 DECODE → CSR
(SCR) PW → I47
3rd DATA F/F (C1 > C2)
CR9 → H → I15 → I5 → C → SN → I5 → C → I30 → Q → S → HR → DATA XMIT
R → I30 → Q
R
CSR → I47
1st DATA · 2nd DATA · 3rd DATA
(SCR) PW → I47
CR9 → H → I15 → I5 → C → SN → I5 → C → I30 → Q → S → HR → DATA XMIT
R → I30 → Q
R
(0) CSR → I47
DT=0 · BF=0 · LAC < 12 F/F
LC → I45 → I40 → I40 DECODE → PW → PW → I47
CR9 → H → I15 → I5 → C → C → I30 → Q → S → HR → DATA XMIT
R → I30 → Q
(0) CSR → I47
DT=0 · BF=1 · LAC < 12 F/F
LC → I45 → I40 → I40 DECODE → PW → I47
CR9 → H → I15 → I5 → C → I30 → Q → S → HR → DATA XMIT

INSTRUCTION: COLLATE TABLE LOOKUP



INSTRUCTION: COMPARE (466 + ~~465~~) 467

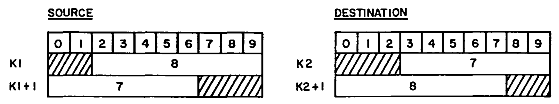


CONDITION
D114
D164
D214
D264
D314
D364
1st DATA · C2 ≥ C1 · TOGGLE
CR9 → H → I15 → I5 → C → SN → I5 → C → I30 → Q → S
C2 → I40 → I40 DECODE → CSR → I47
R
1st DATA · C2 ≥ C1 · TOGGLE
CR9 → H → I15 → I5 → C → I30 → Q
1st DATA · C1 > C2 · TOGGLE
CR9 → H → I15 → I5 → C → I30 → Q → S
C1 → I40 → I40 DECODE → CSR → I47
1st DATA · C1 > C2 · TOGGLE
CR9 → H → I15 → I5 → C → SN → I5 → C → I30 → Q
I47
R
1st DATA · C2 ≥ C1 · LAST COMPARE · TOGGLE
(SCR) PW → I47
CR9 → H → I15 → I5 → C → SN → I5 → C → I30 → Q → S
R → I30 → Q
R
(0) CSR → I47
1st DATA · C2 ≥ C1 · LAST COMPARE · TOGGLE
CR9 → H → I15 → I5 → C → I30 → Q
(0) CSR → I47
1st DATA · C1 > C2 · LAST COMPARE TOGGLE
CR9 → H → I15 → I5 → C → I30 → Q → S
(0) CSR → I47
1st DATA · C1 > C2 · LAST COMPARE · TOGGLE
(SCR) PW → I47
CR9 → H → I15 → I5 → C → SN → I5 → C → I30 → Q
R → I30 → Q
R
(0) CSR → I47
LAST COMPARE · BF=2 · C2 ≥ C1 · TOGGLE
CR9 → H → I15 → I5 → C → SN → I5 → C → I30 → Q → S
R → I30 → Q
LC → I45 → I40 → I40 DECODE → CSR → I47
CSR ⊕ PW → I47
(PW=SCR)
I47
LAST COMPARE · BF=2 · C2 ≥ C1 · TOGGLE
CR9 → H → I15 → I5 → C → I30 → Q
LAST COMPARE · BF=2 · C1 > C2 · TOGGLE
CR9 → H → I15 → I5 → C → Q → S
LA → I45 → I40 → I40 DECODE → CSR → I47
I47
(PW=SCR)
CSR ⊕ PW → I47
LAST COMPARE · BF=2 · C1 > C2 · TOGGLE
R → I30 → Q
CR9 → H → I15 → I5 → C → SN → I5 → C → I30 → Q
LAST COMPARE · BF=1 · C2 ≥ C1
CR9 → H → I15 → I5 → C → C → I30 → Q
R → I30 → Q → S
LA → I45 → I40 → I40 DECODE → CSR → I47
I47
LAST COMPARE · BF=1 · C1 > C2
CR9 → H → I15 → I5 → C → I30 → Q → S
I47
R → I30 → Q
LA → I45 → I40 → I40 DECODE → CSR → I47

CONTROL DATA
CANADIAN DEVELOPMENT DIVISION

DATA SEQUENCE

CODE IDENT.		DWG. NO.	REV
34570	D	19981800	A
FIGURE 5-2-33		PAGE NO. 5-2-78-5	



STEP 1

INSTRUCTION DECODE SEQ

$LC \rightarrow LF$
$C2 \rightarrow LE$

2

START SEQ

$LE + LF \rightarrow LC = 22_8\ (L + C2)$
$C1 - C2 \rightarrow SCR = 1 \quad SCR \rightarrow SK$
$0 \rightarrow CSR$
$SCR \rightarrow PW = 1$
$LA - C1 \rightarrow LA = 21_8\ (L + C1)$
$-12_8 \rightarrow LE$

3

ADDRESS SEQ

$\overline{K1}$ — 1ST ADRS: $K2 + RA \rightarrow F \rightarrow$ ADRS XMIT
$LC - LE\ (22_8 - 12_8 = 10_8)$ $DT + 1 = 1$
$\searrow$ TEST K2 EXHAUST ($LC \leq 0$)

K1 — 2ND ADRS: $K1 + RA \rightarrow F \rightarrow$ ADRS XMIT $\quad DT + 1 = 2$
$LA - LE \rightarrow LA\ (21_8 - 12_8 = 7_8)$ $\quad BF + 1 = 1$
$\searrow$ TEST K1 EXHAUST ($LA \leq 0$)

$\overline{\text{1ST ADRS}}$ $\overline{\text{2ND ADRS}}$: $(K1+1) + RA \rightarrow F \rightarrow$ ADRS XMIT
$LA - LE\ (7_8 - 12_8 =$ EXHAUST) $\quad DT + 1 = 3$, $BF + 1 = 2$
$\searrow$ SET K1 EXHAUST

1ST WRITE: $K2 + RA \rightarrow F$
$\overline{K1}$ — $LC - LE \rightarrow LC\ (22_8 - 12_8 = 10_8)$
$\searrow LF - LE\ (10_8 - 12_8 =$ EXHAUST
$\searrow$ SET $LAC < 12$ F/F
$\rightarrow$ TEST K2 EXHAUST

K1 — $(K2+1) + RA \rightarrow F \rightarrow$ ADRS XMIT $\quad DT + 1 = 2$

$\overline{K1}$ — $(K2+1) + RA \rightarrow F$
$LC - LE\ (10_8 - 12_8 =$ EXHAUST)
$\searrow$ SET K2 EXHAUST

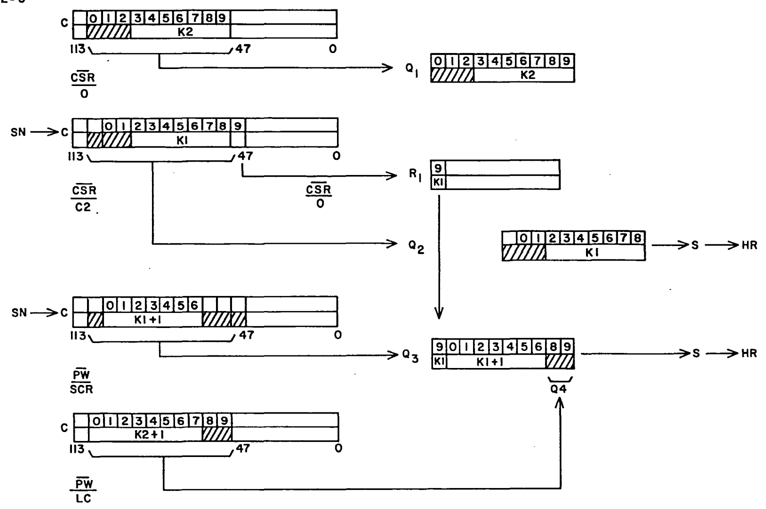
SEQ WILL WAIT UNTIL HR FULL BEFORE ADRS XMIT

SEQ WILL WAIT UNTIL HR FULL BEFORE ADRS XMIT

NOTES: DT : DATA COUNTER
BF : BUFFER COUNTER

MOVE EXAMPLE NO. 1 LONG MOVE $C2 > C1$

$L = 17_8 = 15_{10}$
$C1 = 2$
$C2 = 3$



DATA SEQ

1ST DATA: $K2 \rightarrow CR9 \rightarrow H \rightarrow C \xrightarrow[0]{\overline{CSR}} Q_1$
$DT - 1 = 2$

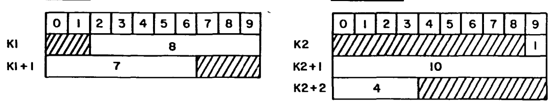
2ND DATA: $K1 \rightarrow CR9 \rightarrow H \rightarrow C \rightarrow SN \rightarrow C \xrightarrow[C2]{\overline{CSR}} Q_2 \rightarrow S \rightarrow HR \rightarrow$ DATA XMIT; $F \rightarrow$ ADRS XMIT; $BF - 1 = 1$
$\searrow R_1$

$K1+1 \rightarrow CR9 \rightarrow H \rightarrow C \rightarrow SN \rightarrow C \xrightarrow[SCR]{\overline{PW}} Q_3 \rightarrow S \rightarrow$ BLOCK HR
$\searrow R_2$
$DT - 1 = 1$
$R_1 \xrightarrow[0]{\overline{CSR}} Q$

$DT = 0 \cdot BF = 0 \cdot LAC < 12$: $K2+1 \rightarrow CR9 \rightarrow H \rightarrow C \xrightarrow[LC]{\overline{PW}} Q_4 \rightarrow S \rightarrow HR \rightarrow$ DATA XMIT; $F \rightarrow$ ADRS XMIT
$DT - 1 = 0$

CONTROL DATA CANADIAN DEVELOPMENT DIVISION	INSTRUCTION FLOW EX I LONG MOVE $C2 > C1$	CODE IDENT 34570 D	DWG NO 19981800 / RE. A
		FIGURE 5-2-34	PAGE NO 5-2-78-6

SOURCE / DESTINATION



INSTRUCTION DECODE SEQ

```
LC ⟶ LF
C2 ⟶ LE
```

START SEQ

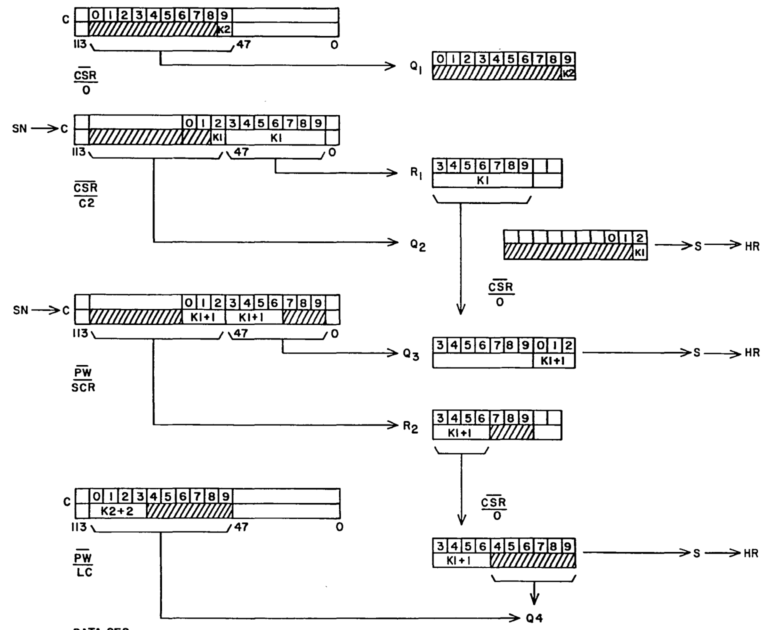
```
LE + LF ⟶ LC = 30₈ (L + C2)
C1 - C2 ⟶ SCR = 7   SCR ⟶ SK
0 ⟶ CSR
SCR ⟶ PW = 7
LA + C1 ⟶ LA = 21₈ (L+C1)
-12₈ ⟶ LE
```

ADDRESS SEQ

```
K1̄ { 1ST ADRS    K2 + RA ⟶ F ⟶ ADRS XMIT
                  LC - LE (30₈ - 12₈ = 16₈) DT + 1 = 1
                     ↘ TEST K2 EXHAUST ( LC ≤ 0 )

K1  { 2ND ADRS    K1 + RA ⟶ F ⟶ ADRS XMIT          DT + 1 = 2
                  LA - LE ⟶ LA (21₈ - 12₈ = 7₈)      BF + 1 = 1
                        ↘ TEST K1 EXHAUST ( LA ≤ 0 )
      1ST ADRS‾   (K1 + 1) + RA ⟶ F ⟶ ADRS XMIT
      2ND ADRS‾   LA - LE (7₈ - 12₈ = EXHAUST)  DT + 1 = 3
                     ↘ SET K1 EXHAUST              BF + 1 = 2

      1ST WRITE   K2 + RA ⟶ F                               ┆ SEQ WILL WAIT UNTIL ┆
K1̄ {              LC - LE ┬⟶ LC (30₈ - 12₈ = 16₈)           ┆ HR FULL BEFORE      ┆
                          ├↘ LF - LE (16₈ - 12₈ = 4)         ┆ ADRS XMIT           ┆
                          │      ↘ TEST ≤ 12 (LF < 12)
                          └⟶ TEST K2 EXHAUST
                  (K2 + 1) + RA ⟶ F                          ┆ SEQ WILL WAIT UNTIL ┆
                  LC - LE ┬⟶ LC (16₈ - 12₈ = 4)              ┆ HR FULL BEFORE      ┆
                          ├↘ LF - LE (4 - 12₈ = EXHAUST)     ┆ ADRS XMIT           ┆
                          │      ↘ SET LAC<12 F/F
                          └⟶ TEST K2 EXHAUST

K1  {             (K2 + 2) + RA ⟶ F ⟶ ADRS XMIT   DT + 1 = 2
K1̄  {             (K2 + 2) + RA ⟶ F                          ┆ SEQ WILL WAIT UNTIL ┆
                  LC - LE (4 - 12₈ = EXHAUST)                ┆ HR FULL BEFORE      ┆
                     ↘ SET K2 EXHAUST                        ┆ ADRS XMIT           ┆
```

MOVE EXAMPLE NO. 2 LONG MOVE C2 > C1

$L = 17_8 = 15_{10}$
C1 = 2
C2 = 9



DATA SEQ

```
1ST DATA        K2 ⟶ CR9 ⟶ H ⟶ C ─CSR‾/0⟶ Q₁
                DT - 1 = 2
                                                                     F ⟶ ADRS XMIT
2ND DATA        K1 ⟶ CR9 ⟶ H ⟶ C ⟶ SN ⟶ C ─CSR‾⟶ Q₂ ⟶ S ⟶ HR ⟶ DATA XMIT
                                             C2 ↘ R₁                       BF - 1 = 1
                                                                     F ⟶ ADRS XMIT
                K1+1 ⟶ CR9 ⟶ H ⟶ C ⟶ SN ⟶ C ─PW‾⟶ Q₃ ⟶ S ⟶ HR ⟶ DATA XMIT
                DT - 1 = 1                  SCR ↘ R₂                      BF - 1 = 0
                               R₁ ─CSR‾/0⟶ Q
                                                                     F ⟶ ADRS XMIT
DT = 0 · BF = 0  K2+2 ⟶ CR9 ⟶ H ⟶ C ──────⟶ C ─PW‾/LC⟶ Q₄ ⟶ S ⟶ HR ⟶ DATA XMIT
· LAC < 12       DT - 1 = 0
                               R₂ ─CSR‾/0⟶ Q
```

CONTROL DATA CANADIAN DEVELOPMENT DIVISION	INSTRUCTION FLOW EX 2 LONG MOVE C2 > C1	CODE IDENT 34570	D	DWG NO 19981800	REV A
		FIGURE 5-2-35		PAGE NO 5-2-78.7	

MOVE

$L = 17_8 = 15_{10}$

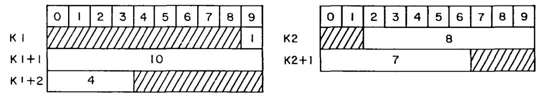
$C1 = 9$

$C2 = 2$

EXAMPLE # 3

LONG MOVE C1 > C2

SOURCE DESTINATION



INSTRUCTION DECODE SEQ

$LC \rightarrow LF$

$C2 \rightarrow LE$

START SEQ

$LE + LF \rightarrow LC = 21_8$ (L + C2)

$(C2 - C1) + 12_8 \rightarrow SCR = 3_8$ $SCR \rightarrow SK$

$0 \rightarrow CSR$

$SCR \rightarrow PW = 3_8$

$LA + C1 \rightarrow LA = 30_8$

$-12_8 \rightarrow LE$

ADDRESS SEQ

$\bar{K}\,\bar{I}$ {
1st ADRS $K2 + RA \rightarrow F \rightarrow$ ADRS XMIT
$LC - LE$ $(21_8 - 12_8 = 7_8)$ $DT + 1 = 1$
$\hookrightarrow$ TEST K2 EXHAUST $(LC \leq 0)$

K1 {
2nd ADRS $K1 + RA \rightarrow F \rightarrow$ ADRS XMIT
$LA - LE \rightarrow LA$ $(30_8 - 12_8 = 16_8)$ $DT + 1 = 2$
$\hookrightarrow$ TEST K1 EXHAUST
$\overline{\text{1st ADRS}} \cdot \text{2nd ADRS}$ $(K1 + 1) + RA \rightarrow F \rightarrow$ ADRS XMIT
$LA - LE \rightarrow LA$ $(30_8 - 12_8 = 4\ )$ $DT + 1 = 3$ $BF + 1 = 1$
$\hookrightarrow$ TEST K1 EXHAUST
$(K1 + 2) + RA \rightarrow F \rightarrow$ ADRS XMIT
$LA - LE$ $(4 - 12_8 =$ EXHAUST) $DT + 1 = 4$ $BF + 1 = 2$
$\hookrightarrow$ SET K1 EXHAUST

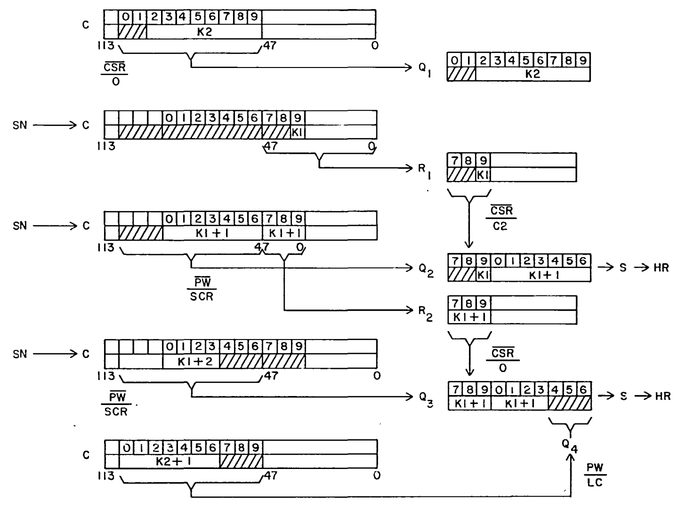
$\bar{K}\,\bar{I}$ {
1st WRITE $K2 + RA \rightarrow F$
$LC - LE \rightarrow LC$ $(21_8 - 12_8 = 7)$
$\hookrightarrow LE - LE$ $(7 - 12_8 =$ EXHAUST)
$\hookrightarrow$ SET LAC < 12 F/F
$\hookrightarrow$ TEST K2 EXHAUST

SEQ WILL WAIT UNTIL HR FULL BEFORE ADRS XMIT

K1 { $(K2 + 1) + RA \rightarrow F \rightarrow$ ADRS XMIT $DT + 1 = 2$

$\overline{K1}$ {
$(K2 + 1) + RA \rightarrow F$
$LC - LE$ $(7 - 12_8 =$ EXHAUST)
$\hookrightarrow$ SET K2 EXHAUST

SEQ WILL WAIT UNTIL HR FULL BEFORE ADRS XMIT



DATA SEQ

1st DATA $K2 \rightarrow CR9 \rightarrow H \rightarrow C \xrightarrow[0]{\overline{CSR}} Q_1$
$DT - 1 = 3$

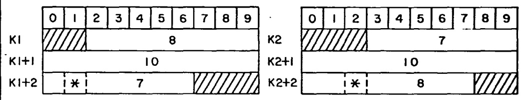
2nd DATA $K1 \rightarrow CR9 \rightarrow H \rightarrow C \rightarrow SN \rightarrow C \rightarrow R$
$DT - 1 = 2$

3rd DATA $K1 + 1 \rightarrow CR9 \rightarrow H \rightarrow C \rightarrow SN \rightarrow C \xrightarrow[SCR]{\overline{PW}} Q_2 \rightarrow S \rightarrow HR \rightarrow$ DATA XMIT; $F \rightarrow$ ADRS XMIT; $BF - 1 = 1$
$C \rightarrow R$
$DT - 1 = 1$
$R \xrightarrow[0]{\overline{CSR}} Q$

$K1 + 2 \rightarrow CR9 \rightarrow H \rightarrow C \rightarrow SN \rightarrow C \xrightarrow[SCR]{\overline{PW}} Q_3 \rightarrow$ BLOCK HR
$DT - 1 = 1$
$R \xrightarrow[0]{\overline{CSR}} Q$

$DT = 0 \cdot BF = 1 \cdot LAC < 12$ $K2 + 1 \rightarrow CR9 \rightarrow H \rightarrow C \xrightarrow[LC]{\overline{PW}} Q_4 \rightarrow S \rightarrow HR \rightarrow$ DATA XMIT; $F \rightarrow$ ADRS XMIT
$DT - 1 = 0$

INSTR FLOW EX: LONG MOVE C1 > C2		19981800	A
		FIGURE 5-2-36	5-2-78.8



* NOTE : THIS EXAMPLE ASSUMES INDICATED CHARACTER OF K1+2 AND K2+2 TO BE UNEQUAL

COMPARE
$L = 31_8 = 25_{10}$
$C1 = 2$
$C2 = 3$

EXAMPLE # 1 $C2 \geq C1$

INSTRUCTION DECODE SEQ

$LA \rightarrow LF$
$C1 \rightarrow LE$

START SEQ

$LE + LF \rightarrow LA = 33_8 \ (L + C1)$
$C2 - C1 \rightarrow SCR = 1 \quad SCR \rightarrow SK$
$0 \rightarrow CSR$
$SCR \rightarrow PW = 1$
$LC + C2 \rightarrow LC = 34_8 \ (L + C2)$
$-12_8 \rightarrow LE$

ADDRESS SEQ

K1 { 1st ADRS
$K1 + RA \rightarrow F \rightarrow$ ADRS XMIT
$LA - LE \rightarrow LA \quad (33_8 - 12_8 = 21_8)$ DT + 1 = 1, BF + 1 = 1
→ TEST K1 EXHAUST

$\overline{K1}$ { 2nd ADRS
$K2 + RA \rightarrow F \rightarrow$ ADRS XMIT
$LC - LE \rightarrow LC \quad (34 - 12_8 = 22_8)$ DT + 1 = 2, BF + 1 = 2
→ TEST K2 EXHAUST
→ LAC1

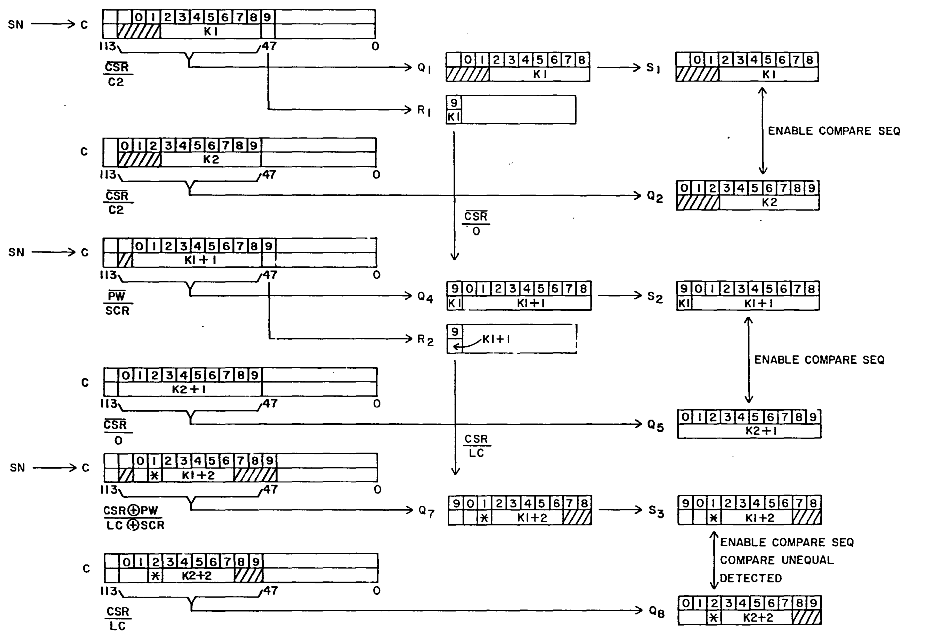
K1 {
$(K1 + 1) + RA \rightarrow F \rightarrow$ ADRS XMIT
$LA - LE \rightarrow LA \quad (21_8 - 12_8 = 7_8)$ DT + 1 = 3, BF + 1 = 3
→ TEST K1 EXHAUST

$\overline{K1}$ {
$(K2 + 1) + RA \rightarrow F \rightarrow$ ADRS XMIT
$LC - LE \rightarrow LC \quad (22_8 - 12_8 = 10_8)$ DT + 1 = 4, BF + 1 = 4
→ TEST K2 EXHAUST
SET BLOCK K ADRS
→ LAC1

ADRS SEQ WILL CONTINUE AFTER BLOCK K ADRS F/F IS CLEARED

K1 {
$(K1 + 2) + RA \rightarrow F \rightarrow$ ADRS XMIT
$LA - LE \quad (7_8 - 12_8 = \text{EXHAUST})$ DT + 1 = 3, BF + 1 = 3
→ SET K1 EXHAUST

$\overline{K1}$ {
$(K2 + 2) + RA \rightarrow F \rightarrow$ ADRS XMIT
$LC - LE \quad (10_8 - 12_8 = \text{EXHAUST}$ DT + 1 = 4, BF + 1 = 4
→ SET K2 EXHAUST
SET BLOCK K ADRS
$LAC1 \rightarrow LAC2 = 22_8$
→ $LC \rightarrow LAC1 = 10_8$



DATA SEQ

1st DATA · C2 ≥ C1 · $\overline{\text{TOGGLE}}$ — $K1 \rightarrow CR9 \rightarrow H \rightarrow C \rightarrow SN \rightarrow C \xrightarrow[C2]{\overline{CSR}} Q_1 \rightarrow S_1$, $\rightarrow R_1$; DT − 1 = 3

1st DATA · C2 ≥ C1 · TOGGLE — $K2 \rightarrow CR9 \rightarrow H \rightarrow C \xrightarrow[C2]{\overline{CSR}} Q_2$; DT − 1 = 3 ENABLE COMPARE

COMPARE SEQ
BF = 4 − 2 = 2 , CLR BLOCK K ADRS

$\overline{\text{1st DATA}}$ · C2 ≥ C1 · $\overline{\text{TOGGLE}}$ — $K1 + 1 \rightarrow CR9 \rightarrow H \rightarrow C \rightarrow SN \rightarrow C \xrightarrow[SCR]{\overline{PW}} Q_4 \rightarrow S_2$, $\rightarrow R_2$; $R \xrightarrow[0]{\overline{CSR}} Q_3$; DT − 1 = 3

$\overline{\text{1st DATA}}$ · C2 ≥ C1 · TOGGLE — $K2 + 1 \rightarrow CR9 \rightarrow H \rightarrow C \xrightarrow[0]{\overline{CSR}} Q_5$; DT − 1 = 3 ENABLE COMPARE

COMPARE SEQ
BF = 4 − 2 = 2 , SET LAST COMPARE F/F , K1 · K2 EXHAUST PREVENTS CLEARING BLOCK K ADRS F/F AND SINCE COMPARE IS EQUAL—ALLOWS $LAC1 \rightarrow LAC2 = 10_8$

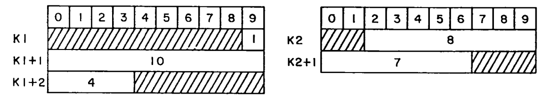
LAST COMPARE · BF = 2 · C2 ≥ C1 · $\overline{\text{TOGGLE}}$ — $K1 + 2 \rightarrow CR9 \rightarrow H \rightarrow C \rightarrow SN \rightarrow C \xrightarrow[LC \oplus SCR]{CSR \oplus PW} Q_T \rightarrow S_3$; $R \xrightarrow[LC]{CSR} Q_6$

LAST COMPARE · BF = 2 · C2 ≥ C1 · TOGGLE — $K2 + 2 \rightarrow CR9 \rightarrow H \rightarrow C \xrightarrow[LC]{CSR} Q_8$; DT − 1 = 0 ENABLE COMPARE.

COMPARE SEQ
BF = 2 − 2 = 0 , UNEQUAL CHARACTER POSITION → CP RGTR = 2 , SELECT UNEQUAL CHARACTER FROM S → TS RGTR, SELECT UNEQUAL CHARACTER FROM Q → TQ RGTR , CLR LAST COMPARE F/F ENABLE EXIT SEQ

EXIT SEQ
$0 \rightarrow C$ RGTR
$(LAC2 - CP) \rightarrow LC \rightarrow I5 \rightarrow C \rightarrow X_0 \qquad (LC = 10_8 - 2_8 = 6_8)$
COMP (IF $\overline{TQ > TS}$)

	INSTR FLOW EX 1 COMPARE C2 > C1		19981800	A
			FIGURE 5-2-37	5-2-78-9



THIS EXAMPLE ASSUMES ALL CHARACTERS EQUAL

INSTRUCTION DECODE SEQ

$LA \rightarrow LF$
$C1 \rightarrow LE$

START SEQ

$LE + LF \rightarrow LA = 30_8$ (L + C1)
$C1 - C2 \rightarrow SCR = 7_8$ $SCR \rightarrow SK$

$SCR \xrightarrow{0 \quad CSR} PW = 7_8$

$LC + C1 \rightarrow LC = 21_8$ (L + C2)
$-12_8 \rightarrow LE$

ADDRESS SEQ

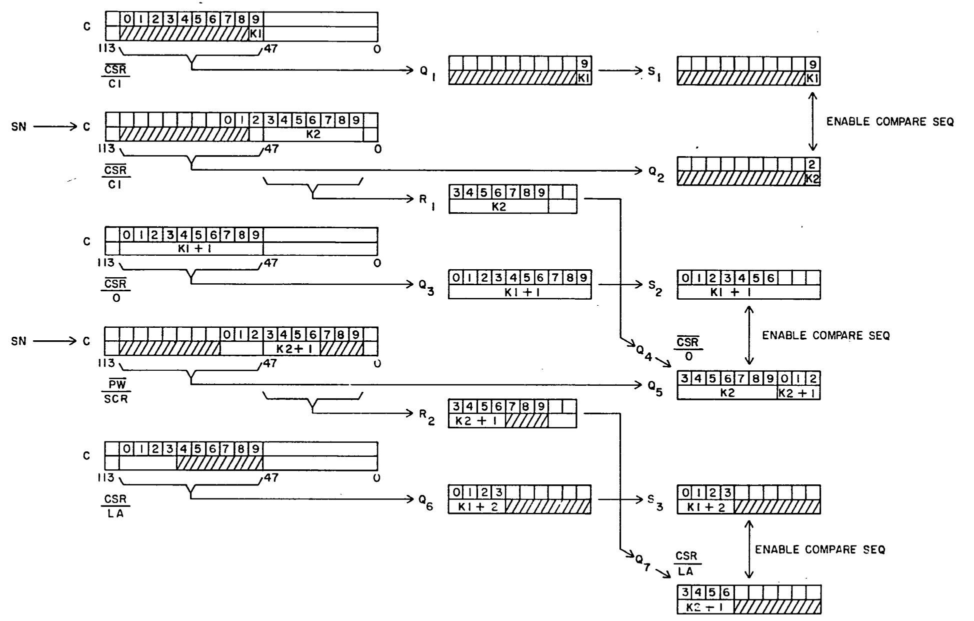
```
K1  { 1st ADRS  K1 + RA → F → ADRS XMIT
                LA - LE → LA   (30₈ - 12₈ = 16₈)   DT + 1 = 1
                 ↘ LAC1 ↘ TEST K1 EXHAUST           BF + 1 = 1

K1‾ { 2nd ADRS  K2 + RA → F → ADRS XMIT
                LC - LE → LC   (21₈ - 12₈ = 7₈)    DT + 1 = 2
                      ↘ TEST K2 EXHAUST             BF + 1 = 2

K1  {           (K1 + 1) + RA → F → ADRS XMIT
                LA - LE → LA   (16₈ - 12₈ = 4))    DT + 1 = 3
                 ↘ LAC1 ↘ TEST K1 EXHAUST           BF + 1 = 3

K1‾ {           (K2 + 1) + RA → F → ADRS XMIT
                LC - LE   (7₈ - 12₈ = EXHAUST)      DT + 1 = 4
                      ↘ SET K2 EXHAUST              BF + 1 = 4
                        SET BLOCK K ADRS F/F  ┌ ADRS SEQ WILL     ┐
                                              │ CONTINUE AFTER    │
                                              │ BLOCK K ADRS      │
                                              └ F/F IS CLEARED    ┘

K1  {           (K1 + 2) + RA → F → ADRS XMIT
                LA - LE   (4 - 12₈ = EXHAUST)       DT + 1 = 3
                │     ↘ SET K1 EXHAUST              BF + 1 = 3
                │       SET BLOCK K ADRS F/F
                │    LAC1 → LAC2 = 16₈
                └──→ LAC1 = 4₈
```

COMPARE

$L = 17_8 = 15_8$
$C1 = 9$
$C2 = 2$

EXAMPLE # 2

$C1 \geq C2$



DATA SEQ

1st DATA $\cdot$ C1 $\geq$ C2 $\cdot$ $\overline{\text{TOGGLE}}$ $K1 \rightarrow CR9 \rightarrow H \rightarrow C \xrightarrow[C1]{\overline{CSR}} Q_1 \rightarrow S_1$
DT - 1 = 3

1st DATA $\cdot$ C1 $\geq$ C2 $\cdot$ TOGGLE $K2 \rightarrow CR9 \rightarrow H \rightarrow C \rightarrow SN \rightarrow C \xrightarrow[C1]{\overline{CSR}} Q_2$ ENABLE COMPARE, $\rightarrow R_1$
DT - 1 = 2

COMPARE SEQ

BF = 4 - 2 = 2 , CLR BLOCK K ADRS F/F

$\overline{\text{1st DATA}}$ $\cdot$ C1 $\geq$ C2 $\cdot$ $\overline{\text{TOGGLE}}$ $K1 + 1 \rightarrow CR9 \rightarrow H \rightarrow C \xrightarrow[0]{\overline{CSR}} Q_3 \rightarrow S_2$
DT - 1 = 2

$\overline{\text{1st DATA}}$ $\cdot$ C1 $\geq$ C2 $\cdot$ TOGGLE $K2 \rightarrow CR9 \rightarrow H \rightarrow C \rightarrow SN \rightarrow C \xrightarrow{\overline{PW}} Q_5$ ENABLE COMPARE, $\xrightarrow{SCR} R_2$
DT - 1 = 1 $R_1 \xrightarrow[0]{\overline{CSR}} Q_4$

COMPARE SEQ

BF = 3 - 2 = 1 , SET LAST COMPARE F/F , K1 $\cdot$ K2 EXHAUST PREVENTS CLEARING BLOCK K ADRS F/F AND SINCE COMPARE IS EQUAL - ALLOWS $LAC1 \rightarrow LAC2 = 4_8$

LAST COMPARE $\cdot$ BF = 1 $\cdot$ C1 $\geq$ C2 $K1 + 2 \rightarrow CR9 \rightarrow H \rightarrow C \xrightarrow[LA]{CSR} Q_6 \longrightarrow S_3$
DT - 1 = 2 $R_2 \xrightarrow[LA]{CSR} Q_7$ ENABLE COMPARE

COMPARE SEQ

LAST COMPARE $\cdot$ COMPARE EQUAL ENABLE EXIT SEQ

EXIT SEQ

$0 \rightarrow C$ RGTR
$C \rightarrow X_0$

INSTR FLOW EX 2 COMPARE C1 > C2		19981800	A
		FIGURE 5-2-38	5-2-78-10

TABLE 5-2-23. COMPARE/MOVE COMMAND TIMING

SEQUENCE: CENTRAL MEMORY CONTROL ACCEPT

TIME	SIGNAL NAME	TEST POINT	P	COMMAND	CONDITION	COMMENTS
	(3.35)			SET CMC ACCEPT FF	CMACM. CMU ON. $\overline{\text{TLSS}}$	CMC ACCEPT
	(3.35)			CLR ADRS XMIT FULL FF		
	(3.37)			DECREMENT BUFFER COUNTER	$\overline{\text{UPBKPV}}$. (CMC ACCEPT. MOVE. $\overline{\text{K1ADRS XMIT FF}}$)	
	EEXNVF (3.37)	N33-1	F	ENABLE EXIT SEQ	ENABLE EXIT FF. K1. K2 EXHAUST. MOVE. CMC ACCEPT. $\overline{\text{K1 ADRS XMIT FF}}$	
	SBHRVK (3.37)	M31-10	F	SET BLOCK HR FF	COMPARE + (K1 ADRS. BUF=2. LAC < 12 FF. $\overline{\text{K1 ADRS XMIT FF}}$. CMC ACCEPT)	
	CHRFSW (3.37)	M30-11	F	CLR HR FULL FF	MOVE. CMC ACCEPT. $\overline{\text{K1 ADRS XMIT FF}}$	

TABLE 5-2-24. COMPARE/MOVE COMMAND TIMING

SEQUENCE: CENTRAL MEMORY CONTROL DATA READY RESPONSE

TIME	SIGNAL NAME	TEST POINT	P	COMMAND	CONDITION	COMMENTS
	CMDRM (3.39)	M32-1	T	GENERATE DATA READY-CMDRJX	DATA READY. CMU ON FF	
	LA45TV (3.41)	M27-4	F	SELECT LA → I45	1st & LAST FF. $\overline{\text{COLLATE IN PROGRESS}}$. COMPARE. C1 ≥ C2. DATA READY DELAY FF	
	$\overline{\text{LA45TV}}$ (3.41)	M27-4	T	SELECT LC → I45		

TABLE 5-2-25. COMPARE/MOVE COMMAND TIMING

SEQUENCE: DATA

Page 1 of 5

TIME	SIGNAL NAME	TEST POINT	P	COMMAND	CONDITION	COMMENTS
D114	CMDRJX (3.39)	M32-10	F	DECREMENT DATA COUNTER	DATA READY DELAY FF $\overline{\text{UPDKPJ}}$. $\overline{\text{DT=0}}$. $\overline{\text{1st \& LAST FF}}$ + COLLATE IN PROGRESS FF	
	ENABH (3.39)			ENABLE H RGTR	DATA READY DELAY FF. ($\overline{\text{1st \& LAST FF}}$ + COLLATE IN PROGRESS. $\overline{\text{DT=0}}$)	
				SET H FULL FF	DATA READY DELAY FF. ($\overline{\text{1st \& LAST FF}}$ + COLLATE IN PROGRESS). $\overline{\text{DT=0}}$	
	ENHBT (3.39)			ENABLE T RGTR	DATA READY DELAY FF. ($\overline{\text{1st \& LAST FF}}$ + COLLATE IN PROGRESS). DT=0	
				SET T FULL FF	DATA READY DELAY FF. ($\overline{\text{1st \& LAST FF}}$ + COLLATE IN PROGRESS). DT=0	
	E164JW (3.39)	M30-13	T	ENABLE D164	DATA READY DELAY FF. ($\overline{\text{1st \& LAST FF}}$ + COLLATE IN PROGRESS). DT=0	
	LMOOMK (3.40)	M31-5	T		LAST MOVE. DT0. LAC < 12. (BUF. 0 + BUF=1)	
	LCOOMN (3.40)	M29-6)	T		LAST COMPARE. LAST WORD COMPARE FF. (BUF=1 + BUF. 2)	

TABLE 5-2-25. COMPARE/MOVE COMMAND TIMING

SEQUENCE: DATA (cont.)

Page 2 of 5

TIME	SIGNAL NAME	TEST POINT	P	COMMAND	CONDITION	COMMENTS
				ENABLE D164	E164JW + E164KW + (C FULL FF. H FULL FF. D214) + (H FULL FF. $\overline{\text{C FULL FF}}$)	
D164				SELECT H → I15		
	15I5JT (3.39)			SELECT I15 → I5	$\overline{\text{T FULL FF}}$ • $\overline{\text{BLK C FULL}}$	
	SCFOJW (3.39)			SET C FULL FF	$\overline{\text{T FULL FF}}$ • $\overline{\text{BLK C FULL}}$	
				CLR H FULL FF	$\overline{\text{T FULL FF}}$	
	ECD114 (3.39)			ENABLE C RGTR	$\overline{\text{T FULL FF}}$	
	C240KL (3.38)	M26-6	F	SELECT C2 → I40	COMPARE. 1st DATA FF. C2 ≥ C1 + 2nd DATA FF	
	ESRDKF (3.38)			ENABLE CSR	COMPARE. 1st DATA FF + 2nd DATA FF	
	C140KL (3.38)	M26-11	F	SELECT C1 → I40	COMPARE. 1st DATA FF. C1 ≥ C2	
	$\overline{\text{G450}}.\overline{\text{G451}}$ (3.37)			SELECT LC → I45		NORMALLY SELECTED
	$\overline{\text{G400}}.\overline{\text{G401}}$ (3.30)			SELECT I45 → I40		NORMALLY SELECTED
	EPWNKF (3.38)			ENABLE PW RGTR	LMOOMK	
	TS38KC (3.38)	L26-11	F	SELECT TS → I38 ENABLE TS RGTR	T FULL FF. (DATA 2 FF + TQ= WP)	
	$\overline{\text{TS38KC}}$ (3.38)	L26-11	T	SELECT TQ → I38	T FULL FF. ($\overline{\text{DATA 2 FF}}$. $\overline{\text{TQ= WP}}$)	NORMALLY SELECTED
	ETQNKF (3.38)	L25-1	F	ENABLE TQ RGTR		
	ECS2KZ (3.38)	L27-7	F	ENABLE COLLATE CS264	T FULL FF. $\overline{\text{DATA 2 FF}}$	
	CTFOKJ (3.38)	M32-4	F	CLR T FULL FF	T FULL FF. ($\overline{\text{DATA 2 FF}}$ + $\overline{\text{TQ=TS}}$)	
	E164KW (3.38)	M30-14	T	ENABLE DATA SEQ D164	T FULL FF. DATA 2 FF. TS=TQ	
	CDA2KZ (3.38)	L27-8	F	CLR DATA 2 FF	T FULL FF. DATA 2 FF	
	$\overline{\text{G372}}$ (3.42)	M19-10	F	SELECT I34 → I37		NORMALLY SELECTED

TABLE 5-2-25. COMPARE/MOVE COMMAND TIMING

SEQUENCE: DATA (cont.)

Page 3 of 5

TIME	SIGNAL NAME	TEST POINT	P	COMMAND	CONDITION	COMMENTS
	D214D (3.38)			ENABLE D214	C FULL.$\overline{\text{Q FULL}}$.$\overline{\text{D214}}$ + C FULL.Q FULL.$\overline{\text{S FULL}}$ + C FULL.Q FULL.S FULL.$\overline{\text{HR FULL}}$	
D214	EQ00MF (3.40)			ENABLE Q RGTR	MOVE.$\overline{\text{2ndDATA FF}}$ + COMPARE.$\overline{\text{1st DATA FF}}$ + COMPARE.(C1>C2.$\overline{\text{TOGGLE FF}}$ + C2≥C1.TOGGLE FF)	
	ECSNMU (3.40)	L30-1	F	SELECT SN → I5 ENABLE C RGTR	MOVE.$\overline{\text{LM00MK}}$ + COMPARE.(C2≥C1.$\overline{\text{TOGGLE FF}}$ + C1>C2.TOGGLE FF.($\overline{\text{LAST COMPARE FF}}$ + $\overline{\text{BUF1VM}}$)	LM00MK=LAST MOVE (3.40)
	R30DMF (3.40)			SELECT C → I30	MOVE.1st DATA + (LAC<12 FF.DT=0.BUF1VM) + COMPARE.(C2≥C1.TOGGLE FF + C1>C2.$\overline{\text{TOGGLE FF}}$)	
	R30DMF (3.40)			SELECT R → I30	MOVE.$\overline{\text{1st DATA}}$.($\overline{\text{LAC<12 FF}}$ + $\overline{\text{DT=0}}$ + $\overline{\text{BUF1VM}}$) + COMPARE.(C2≥C1.$\overline{\text{TOGGLE FF}}$ + C1>C2.TOGGLE FF)	
	SR47MN (3.40)	M29-1	F	SELECT CSR → I47	COMPARE.LC00MN	LC00MN=LAST COMPARE (3.40)
	CSFDMC (3.40)	L28-10	F	SELECT $\overline{\text{PW}}$ → I47	MOVE.LAC<12 FF.DT=0.BUF1VM	
	$\overline{\text{SR47MN}}$ + $\overline{\text{CSFDMC}}$			SELECT $\overline{\text{CSR}}$ → I47	MOVE.($\overline{\text{LAC<12 FF}}$ + $\overline{\text{DT=0}}$ + $\overline{\text{BUF1VM}}$) + COMPARE.$\overline{\text{LC00MN}}$	
	EC0MMF (3.39)	M33-7	F	ENABLE COMPARE CM114	COMPARE.C2≥C1.TOGGLE FF	
	ETOGMN (3.40)	M29-11	F	ENABLE TOGGLE FF	COMPARE.(C2≥C1.TOGGLE FF + C1>C2.$\overline{\text{TOGGLE FF}}$)	
	SQF0MW (3.40)			ENABLE SET Q FULL FF	MOVE.LAC<12 FF.DT=0.BUF1VM + COMPARE.(C1>C2.TOGGLE FF + C2≥C1.TOGGLE FF) + LAST COMPARE FF.BUF1VM	
	(3.38)	M30-2	F	SET Q FULL FF	SQF0MW.$\overline{\text{3rd DATA FF}}$	
	CCF0MW (3.40)	M30-4)	F	CLR C FULL FF	MOVE.(1st DATA FF + LAC<12 FF.DT=0.BUF1VM) + COMPARE.(C1>C2.$\overline{\text{TOGGLE FF}}$ + C2≥C1.TOGGLE FF)	
	CSFDMC (3.40)	L28-10	F	CLR S FULL FF	MOVE.LAC 12 FF.DT=0.BUF1VM	
	(3.38)			CLR BLOCK HR FF	LM00MK	LM00MK=LAST MOVE (3.40)
	C1S2MN (3.40)	M29-10	F	CLR 1st DATA FF SET 2nd DATA FF	1st DATA.MOVE	
	ECSRMC (3.40)	N29-11	F	SELECT 0 → I40 DECODER ENABLE CSR RGTR CLR 1st DATA FF	COMPARE.C2≥C1.TOGGLE FF.1st DATA FF	

TABLE 5-2-25. COMPARE/MOVE COMMAND TIMING

SEQUENCE: DATA (cont.)

Page 4 of 5

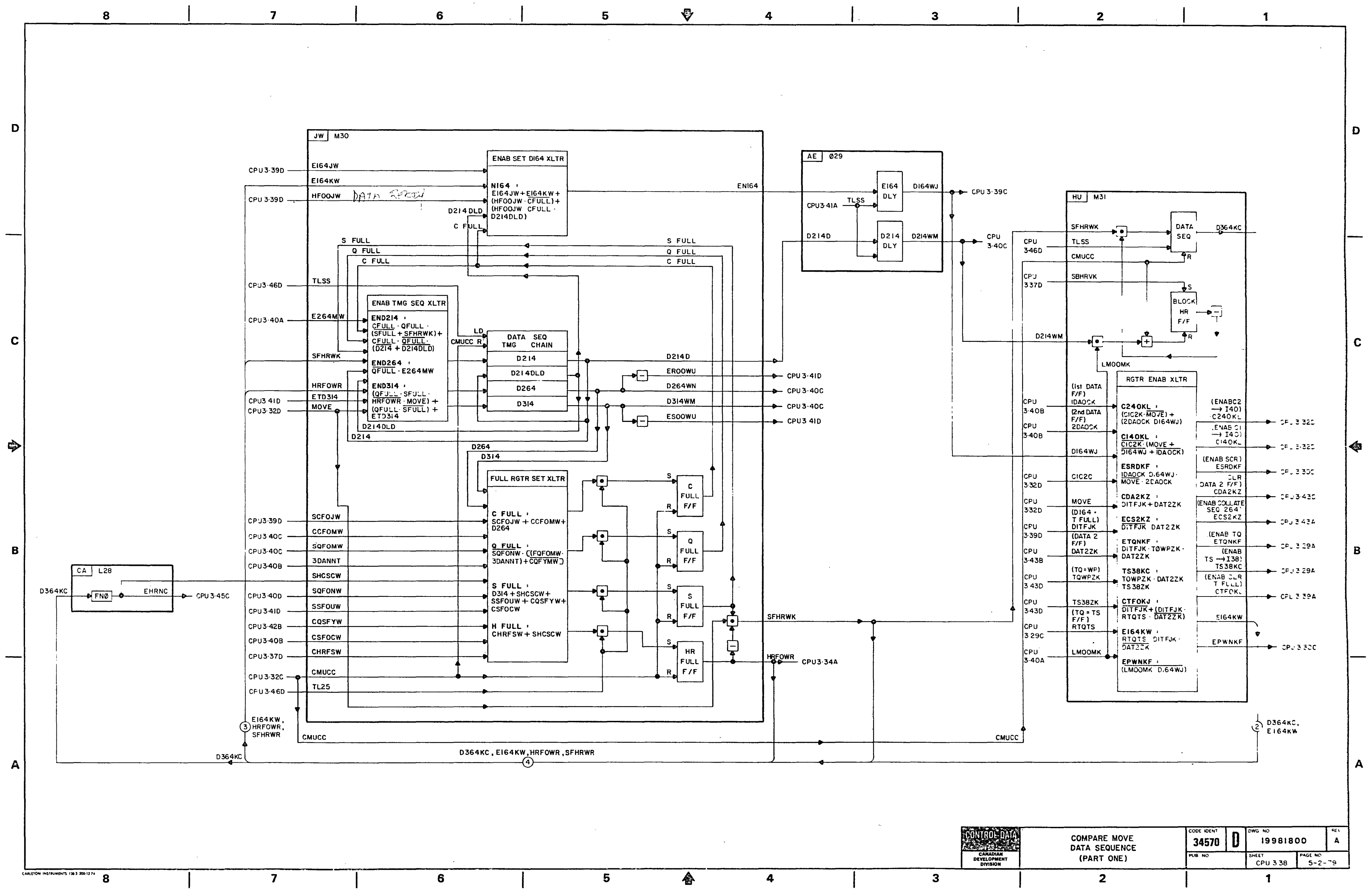
TIME	SIGNAL NAME		TEST POINT	P	COMMAND	CONDITION	COMMENTS
	E264MW	(3.40)			ENABLE D264	[MOVE.$\overline{\text{1st DATA FF}}$ + COMPARE] . D214	
		(3.38)			SET D264	E264MW.$\overline{\text{Q FULL FF}}$	
D264	ER00WU	(3.38)			ENABLE R RGTR		
	$\overline{\text{G302}}$	(3.28)			SELECT C → I30		$\overline{\text{R → I30}}$
		(3.38)			CLR C FULL FF		
	ETOGUN	(3.41)	M29-14	F	ENABLE TOGGLE FF		
	EQ00NF	(3.40)			ENABLE Q RGTR	(2nd DATA FF.C2 ≥C1) + $\overline{\text{2nd DATA FF}}$	
	G470.G471	(3.40)			SELECT PW ≠ CSR → I47	LC00MN	LC00MN=LAST COMPARE (2.39)
	$\overline{\text{G470}}$.$\overline{\text{G471}}$	(3.40)			SELECT PW → I47	MOVE.($\overline{\text{2nd DATA FF}}$ + C1 ≥C2) + COMPARE .$\overline{\text{1st DATA}}$.$\overline{\text{LC00MN}}$ + SR47MN + PW47CN	
	G470.$\overline{\text{G471}}$	(3.40)			SELECT CSR → I47	MOVE.(2nd DATA FF.C2 ≥C1) + COMPARE .1st DATA + LC00MN	$\overline{\text{PW → I47}}$ + $\overline{\text{PW}}$≠$\overline{\text{CSR → I47}}$
	EC0MNF	(3.40)			ENABLE COMPARE SEQ	COMPARE.TOGGLE FF	
	SQF0NW	(3.40)	M30-3	F	SET Q FULL FF	C2 ≥ C1 + $\overline{\text{2nd DATA FF}}$	
	ECSRNC	(3.40)			SELECT 0 → I40 DECODE / ENABLE CSR RGTR	2nd DATA FF.C2 ≥C1 + 3rd DATA FF + COMPARE.1st DATA FF.TOGGLE FF	
					CLR 1st DATA FF	COMPARE.1st DATA FF.TOGGLE FF	
					CLR 2nd DATA FF	2nd DATA FF	
					CLR 3rd DATA FF	3rd DATA FF	
					SET 3rd DATA FF	2nd DATA FF.C1 ≥C2	

TABLE 5-2-25. COMPARE/MOVE COMMAND TIMING

SEQUENCE: DATA (cont.)

Page 5 of 5

TIME	SIGNAL NAME	TEST POINT	P	COMMAND	CONDITION	COMMENTS
	(3.38)			ENABLE D314	Q FULL.$\overline{\text{S FULL}}$ + Q FULL.S FULL.$\overline{\text{HR FULL}}$.MOVE + ETD314	
D314	(3.38)			SET S FULL FF	SSFOUW	
	ES00WU (3.38)	L30-3	F	ENABLE S RGTR		
	(3.38)			CLR Q FULL FF		
	$\overline{\text{R30DMF}}$ (3.40)			SELECT R → I30	LAST WORD COMPARE FF.BUF1VM.C1 ≥C2	
	SR47MN (3.40)	M29-1	F	SELECT CSR → I47	LAST WORD COMPARE FF.BUF1VM	
	EQ00MF (3.40)			ENABLE Q RGTR	LAST WORD COMPARE FF.BUF1VM	
	ECOMMF (3.40)	N33-7	F	ENABLE COMPARE SEQ	LAST WORD COMPARE FF.BUF1VM	
	SFHRWK (3.38)	M31-9	T	ENABLE D364	S FULL.$\overline{\text{HR FULL}}$.MOVE	
	(3.38)			SET D364	SFHRWK.$\overline{\text{BLOCK HR FF}}$	
D364	EHRNC (3.38)			ENABLE HR RGTR		
	CMI7.CMX7 (3.45)			SELECT S → I7		
	SHCSCW (3.38)	M30-10	F	SET HR FULL FF		
				CLR S FULL FF		

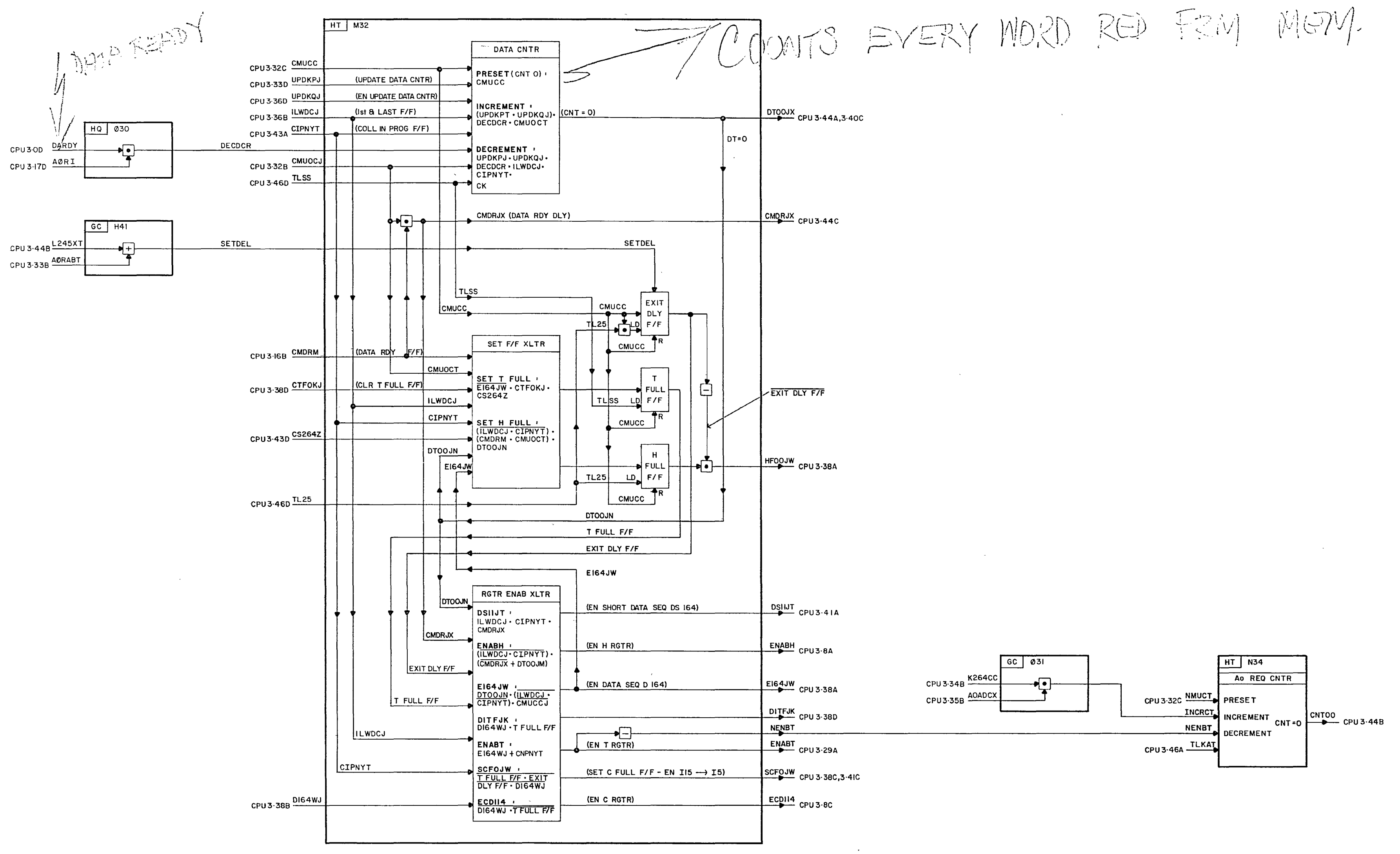


CARLETON INSTRUMENTS

CONTROL DATA
CANADIAN DEVELOPMENT DIVISION

COMPARE MOVE
DATA SEQUENCE
(PART ONE)

CODE IDENT		DWG NO	REV
34570	D	19981800	A
PUB NO		SHEET CPU 3 38	PAGE NO 5-2-79



DATA READY
COUNTS EVERY WORD RED FRM MEM.
HT M32
DATA CNTR
PRESET(CNT 0) · CMUCC
INCREMENT · (UPDKPT · UPDKQJ) · DECDCR · CMUOCT
DECREMENT · UPDKPJ · UPDKQJ · DECDCR · ILWDCJ · CIPNYT ·
CK
(CNT = 0)
DTOOJX CPU 3·44A,3·40C
DT=0
CPU 3·32C CMUCC
CPU 3·33D UPDKPJ (UPDATE DATA CNTR)
CPU 3·36D UPDKQJ (EN UPDATE DATA CNTR)
CPU 3·36B ILWDCJ (1st & LAST F/F)
CPU 3·43A CIPNYT (COLL IN PROG F/F)
CPU 3·32B CMUOCJ
CPU 3·46D TLSS
HQ Ø30
CPU 3·0D DARDY
CPU 3·17D AØRI
DECDCR
GC H41
CPU 3·44B L245XT
CPU 3·33B AØRABT
SETDEL
CMDRJX (DATA RDY DLY)
CMDRJX CPU 3·44C
TLSS
CMUCC
EXIT DLY F/F
TL25 LD
R
T FULL F/F
TLSS LD
H FULL F/F
TL25 LD
EXIT DLY F/F
HFOOJW CPU 3·38A
SET F/F XLTR
CPU 3·16B CMDRM (DATA RDY F/F)
CMUOCT
CPU 3·38D CTFOKJ (CLR T FULL F/F)
ILWDCJ
CIPNYT
CPU 3·43D CS264Z
DTOOJN
E164JW
SET T FULL · E164JW · CTFOKJ · CS264Z
SET H FULL · (ILWDCJ · CIPNYT) · (CMDRM · CMUOCT) · DTOOJN
CPU 3·46D TL25
DTOOJN
T FULL F/F
EXIT DLY F/F
E164JW
RGTR ENAB XLTR
DS11JT · ILWDCJ · CIPNYT · CMDRJX
ENABH · (ILWDCJ · CIPNYT) · (CMDRJX + DTOOJM)
E164JW · DTOOJN · (ILWDCJ · CIPNYT) · CMUCCJ
DITFJK · D164WJ · T FULL F/F
ENABT · E164WJ + CNPNYT
SCFOJW · T FULL F/F · EXIT DLY F/F · D164WJ
ECD114 · D164WJ · T FULL F/F
CMDRJX
EXIT DLY F/F
T FULL F/F
ILWDCJ
CIPNYT
CPU 3·38B D164WJ
(EN SHORT DATA SEQ DS 164) DS11JT CPU 3·41A
(EN H RGTR) ENABH CPU 3·8A
(EN DATA SEQ D 164) E164JW CPU 3·38A
DITFJK CPU 3·38D
NENBT
(EN T RGTR) ENABT CPU 3·29A
(SET C FULL F/F - EN I15 ⟶ I5) SCFOJW CPU 3·38C,3·41C
(EN C RGTR) ECD114 CPU 3·8C
GC Ø31
CPU 3·34B K264CC
CPU 3·35B AOADCX
HT N34
Ao REQ CNTR
CPU 3·32C NMUCT PRESET
INCRCT INCREMENT
NENBT DECREMENT
CPU 3·46A TLKAT
CNT=0
CNTOO CPU 3·44B

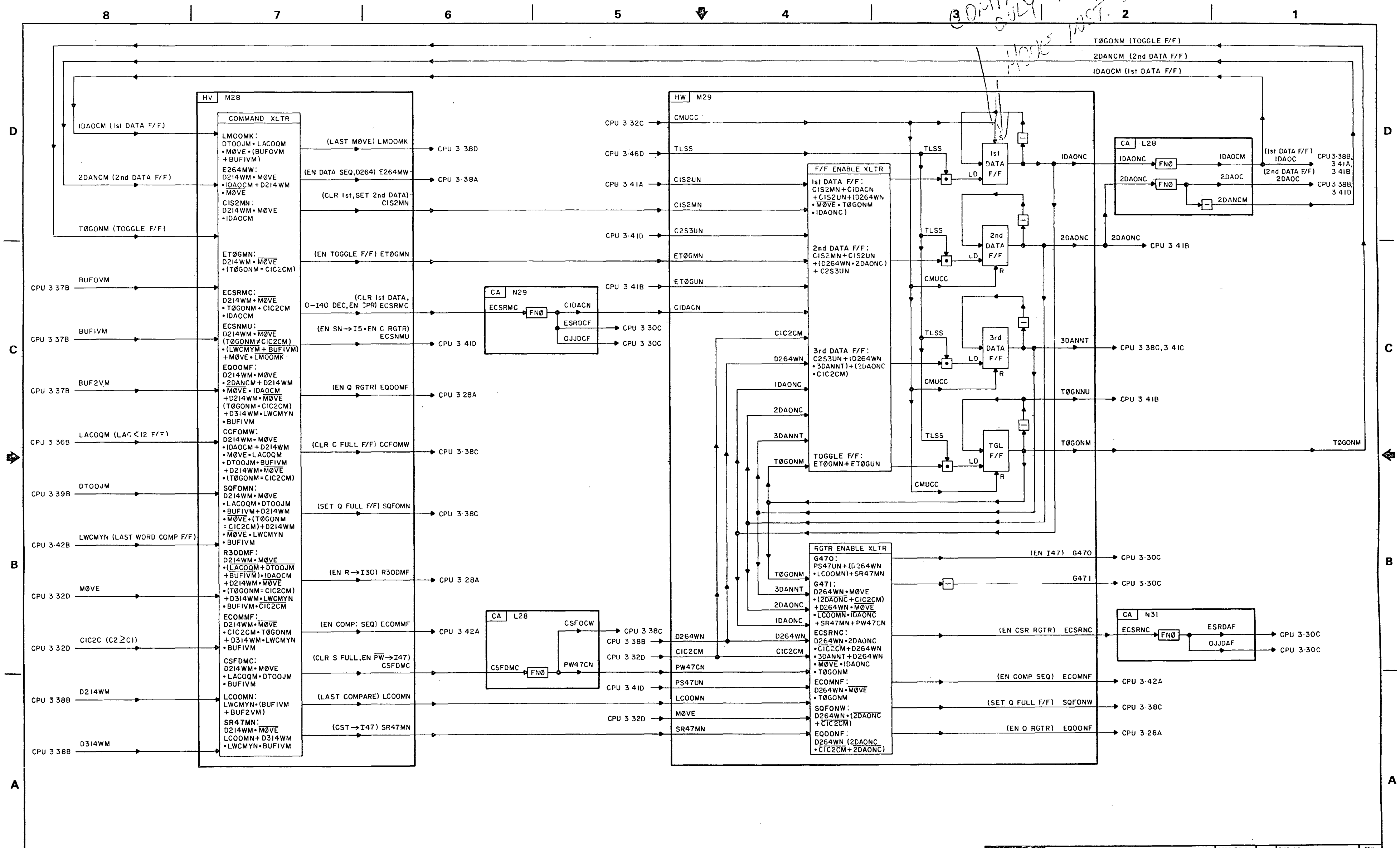
CONTROL DATA CANADIAN DEVELOPMENT DIVISION

COMPARE MOVE DATA SEQUENCE (PART TWO)

CODE IDENT.		DWG. NO.	REV
34570	D	19981800	K
PUB. NO.		SHEET CPU 3·39	PAGE NO. 5-2-81

CARLETON INSTRUMENTS 138-3 200-12-74

COMPARE INST. USES ONLY 1st F/F
MOVE INST. USES ALL F/F



TØGONM (TOGGLE F/F)
2DANCM (2nd DATA F/F)
IDAOCM (1st DATA F/F)
HV M28
COMMAND XLTR
LMOOMK: DTOOJM•LACOQM •MØVE•(BUFOVM +BUFIVM)
E264MW: D214WM•MØVE •IDAOCM+D214WM •MØVE
CIS2MN: D214WM•MØVE •IDAOCM
ETØGMN: D214WM•MØVE •(TØGONM=CIC2CM)
ECSRMC: D214WM•MØVE •TØGONM•CIC2CM •IDAOCM
ECSNMU: D214WM•MØVE (TØGONM≠CIC2CM) •(LWCMYM+BUFIVM) +MØVE•LMOOMK
EQOOMF: D214WM•MØVE •2DANCM+D214WM •MØVE•IDAOCM +D214WM•MØVE (TØGONM=CIC2CM) +D314WM•LWCMYN •BUFIVM
CCFOMW: D214WM•MØVE •IDAOCM+D214WM •MØVE•LACOQM •DTOOJM•BUFIVM +D214WM•MØVE •(TØGONM=CIC2CM)
SQFOMN: D214WM•MØVE •LACOQM•DTOOJM •BUFIVM+D214WM •MØVE•(TØGONM =CIC2CM)+D214WM •MØVE•LWCMYN •BUFIVM
R3ODMF: D214WM•MØVE •(LACOQM+DTOOJM +BUFIVM)•IDAOCM +D214WM•MØVE •(TØGONM=CIC2CM) +D314WM•LWCMYN •BUFIVM•CIC2CM
ECOMMF: D214WM•MØVE •CIC2CM•TØGONM +D314WM•LWCMYN •BUFIVM
CSFDMC: D214WM•MØVE •LACOQM•DTOOJM •BUFIVM
LCOOMN: LWCMYN•(BUFIVM +BUF2VM)
SR47MN: D214WM•MØVE LCOOMN+D314WM •LWCMYN•BUFIVM
(LAST MØVE) LMOOMK → CPU 3 38D
(EN DATA SEQ,D264) E264MW → CPU 3 38A
(CLR 1st,SET 2nd DATA) CIS2MN
(EN TOGGLE F/F) ETØGMN
(CLR 1st DATA, 0-I40 DEC,EN SPR) ECSRMC
(EN SN→I5•EN C RGTR) ECSNMU → CPU 3 41D
(EN Q RGTR) EQOOMF → CPU 3 28A
(CLR C FULL F/F) CCFOMW → CPU 3 38C
(SET Q FULL F/F) SQFOMN → CPU 3 38C
(EN R→I30) R3ODMF → CPU 3 28A
(EN COMP SEQ) ECOMMF → CPU 3 42A
(CLR S FULL,EN PW→I47) CSFDMC
(LAST COMPARE) LCOOMN
(CST→I47) SR47MN
CPU 3 37B BUFOVM
CPU 3 37B BUFIVM
CPU 3 37B BUF2VM
CPU 3 36B LACOQM (LAC<12 F/F)
CPU 3 39B DTOOJM
CPU 3 42B LWCMYN (LAST WORD COMP F/F)
CPU 3 32D MØVE
CPU 3 32D CIC2C (C2≥C1)
CPU 3 38B D214WM
CPU 3 38B D314WM
CA N29 ECSRMC FNØ CIDACN ESRDCF OJJDCF → CPU 3 30C
CA L28 CSFDMC FNØ CSFOCW PW47CN → CPU 3 38C
HW M29
CPU 3 32C CMUCC
CPU 3 46D TLSS
CPU 3 41A CIS2UN
CIS2MN
CPU 3 41D C2S3UN
ETØGMN
CPU 3 41B ETØGUN
CIDACN
F/F ENABLE XLTR
1st DATA F/F: CIS2MN+CIDACN +CIS2UN+(D264WN •MØVE•TØGONM •IDAONC)
2nd DATA F/F: CIS2MN+CIS2UN +(D264WN•2DAONC) +C2S3UN
3rd DATA F/F: C2S3UN+(D264WN •3DANNT)+(2DAONC •CIC2CM)
TOGGLE F/F: ETØGMN+ETØGUN
1st DATA F/F
2nd DATA F/F
3rd DATA F/F
TGL F/F
IDAONC
2DAONC → CPU 3 41B
3DANNT → CPU 3 38C,3 41C
TØGNNU → CPU 3 41B
TØGONM
CA L28 IDAONC FNØ IDAOCM (1st DATA F/F) IDAOC → CPU 3 38B, 3 41A, 3 41B
2DAONC FNØ 2DAOC (2nd DATA F/F) 2DAOC → CPU 3 38B, 3 41D
2DANCM
RGTR ENABLE XLTR
G470: PS47UN+(D264WN •LCOOMN)+SR47MN
G471: D264WN•MØVE •(2DAONC+CIC2CM) +D264WN•MØVE •LCOOMN•IDAONC +SR47MN+PW47CN
ECSRNC: D264WN•2DAONC •CIC2CM+D264WN •3DANNT+D264WN •MØVE•IDAONC •TØGONM
ECOMNF: D264WN•MØVE •TØGONM
SQFONW: D264WN•(2DAONC +CIC2CM)
EQOONF: D264WN (2DAONC •CIC2CM+2DAONC)
(EN I47) G470 → CPU 3 30C
G471 → CPU 3 30C
(EN CSR RGTR) ECSRNC
CA N31 ECSRNC FNØ ESRDAF OJJDAF → CPU 3 30C
(EN COMP SEQ) ECOMNF → CPU 3 42A
(SET Q FULL F/F) SQFONW → CPU 3 38C
(EN Q RGTR) EQOONF → CPU 3 28A
CPU 3 38B D264WN
CPU 3 32D CIC2CM
PW47CN
CPU 3 41D PS47UN
LCOOMN
CPU 3 32D MØVE
SR47MN

CONTROL DATA
CANADIAN DEVELOPMENT DIVISION

COMPARE MOVE DATA SEQUENCE
(PART THREE)

CODE IDENT		DWG NO	REV
34570	D	19981800	A
PUB NO		SHEET	PAGE NO
		CPU 3-40	5-2-83

CARLETON INSTRUMENTS 138 3 200-12 74

DETAILED PAK DIAGRAM (CPU 3.41)

SHORT DATA SEQUENCE

The short data sequence is similar in operation to the normal data sequence; however, it is only enabled when the 1st & last FF is set.

1st & last FF is set for a move when K2 exhausts during 1st address, and for a compare when K1 and K2 exhaust during 2nd address.

MOVE INSTRUCTION (464, 465 - Refer to figure 5-2-39.)

1st DATA

1. Decrement data counter by one.
2. The first K2 word is transferred to Q.
3. Clear 1st data, set 2nd data.

2nd DATA

Path selection for 2nd data is determined by C2 ≥ C1 and the buffer count.

C2 ≥ C1

With C2 ≥ C1, the last move equals-zero signal (LMS0TV) selects the appropriate data path.

1. Decrement data counter by one.
2. The shifted K1 word is transferred to Q. The C2 offset in CSR and the shift count in PW control the loading of Q. CSR ≠ PW ensures that only the shifted characters from K1 are stored in Q, while the K2 offset, and characters not part of the K2 field, are protected. (An example of this condition is illustrated in figure 5-2-41.)
3. The complete word in Q is transferred to S and HR in preparation for the 1st write request.

C1 > C2

With C1 > C2, the buffer count is checked. If the count equals zero, the last move equals-zero signal (LMS0TV) is generated. The sequence follows the same path described for C2 ≥ C1.

If the buffer count equals one, the last move equals-one signal (LMS1TU) is generated. Last move equals-one indicates that two K1 words must be received. (An example of this condition is illustrated in figure 5-2-40.)

1. Decrement data counter by one.
2. After K1 has been shifted, all K1 characters will reside in the residue portion of C. Therefore, the residue must first be transferred to R and then to Q. The C2 offset in CSR and the shift count in PW control the loading of Q.
3. Clear 2nd data, set 3rd data.

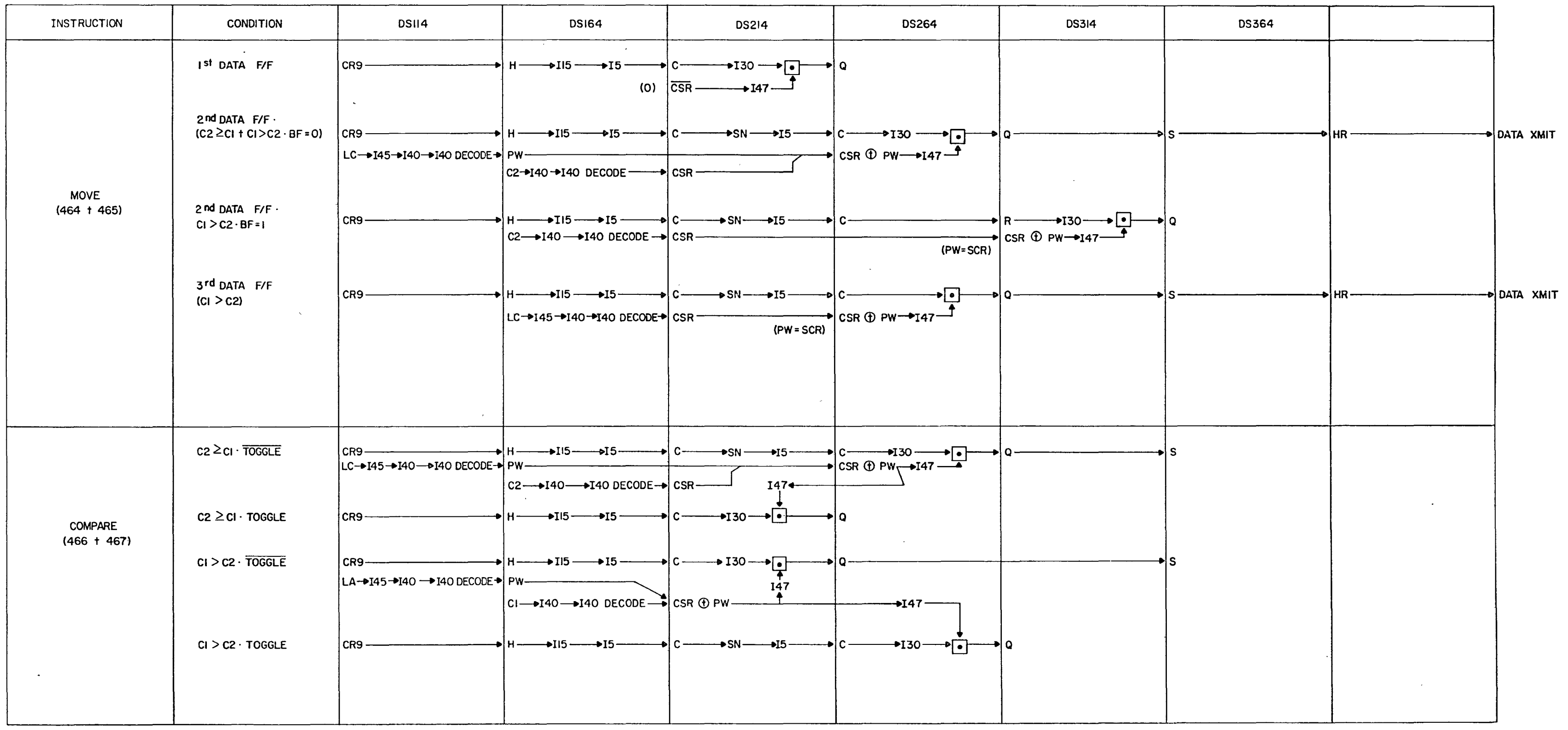
3rd DATA

1. Decrement data counter by one.
2. The shifted K1 data is transferred to Q. The loading of Q is controlled by the remaining length value in LC and the shift count in PW.
3. The complete word in Q is transferred to S and HR in preparation for the 1st write.

COMPARE INSTRUCTION (466, 467 - Refer to figure 5-2-39)

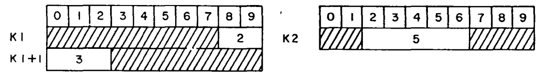
Short data for a compare monitors C2 ≥ C1 and toggle to determine path selection.

The data path is the same as the one used for a normal data sequence with 1st data. except that for a short compare both CSR and PW are used. CSR will contain the C2 offset value (C2 ≥ C1) or the C1 offset value, while PW will contain the remaining LC value (C2 ≥ C1) or LA value (C1 > C2).


INSTRUCTION
CONDITION
DS114
DS164
DS214
DS264
DS314
DS364
MOVE (464 † 465)
1st DATA F/F
CR9 → H → I15 → I5 → C → I30 → ⊡ → Q
(O) CSR → I47
2nd DATA F/F · (C2 ≥ C1 † C1 > C2 · BF = 0)
CR9 → H → I15 → I5 → C → SN → I5 → C → I30 → ⊡ → Q → S → HR → DATA XMIT
LC → I45 → I40 → I40 DECODE → PW → CSR ⊕ PW → I47
C2 → I40 → I40 DECODE → CSR
2nd DATA F/F · C1 > C2 · BF = 1
CR9 → H → I15 → I5 → C → SN → I5 → C → R → I30 → ⊡ → Q
C2 → I40 → I40 DECODE → CSR → CSR ⊕ PW → I47
(PW = SCR)
3rd DATA F/F (C1 > C2)
CR9 → H → I15 → I5 → C → SN → I5 → C → ⊡ → Q → S → HR → DATA XMIT
LC → I45 → I40 → I40 DECODE → CSR → CSR ⊕ PW → I47
(PW = SCR)
COMPARE (466 † 467)
C2 ≥ C1 · TOGGLE (overbar)
CR9 → H → I15 → I5 → C → SN → I5 → C → I30 → ⊡ → Q → S
LC → I45 → I40 → I40 DECODE → PW → CSR ⊕ PW → I47
C2 → I40 → I40 DECODE → CSR
I47
C2 ≥ C1 · TOGGLE
CR9 → H → I15 → I5 → C → I30 → ⊡ → Q
C1 > C2 · TOGGLE (overbar)
CR9 → H → I15 → I5 → C → I30 → ⊡ → Q → S
LA → I45 → I40 → I40 DECODE → PW
C1 → I40 → I40 DECODE → CSR ⊕ PW → I47
I47
C1 > C2 · TOGGLE
CR9 → H → I15 → I5 → C → SN → I5 → C → I30 → ⊡ → Q

SHORT DATA SEQUENCE		19981800	A
		FIGURE 5-2-39	5-2-84·1

SOURCE   DESTINATION



INSTRUCTION DECODE SEQ

$LC \rightarrow LF$
$C2 \rightarrow LE$

START SEQ

$LE + LF \rightarrow LC = 7_8$ $(L + C2)$
$(C2 - C1) + 12_8 \rightarrow SCR = 4_8$ $SCR \rightarrow SK$
$0 \rightarrow CSR$
$SCR \rightarrow PW = 4_8$
$LA + C1 \rightarrow LA = 15_8$
$-12_8 \rightarrow LE$

ADDRESS SEQ

$\overline{K1}$ { 1st ADRS $K2 + RA \rightarrow F \rightarrow$ ADRS XMIT
$LC - LE$ $(7_8 - 12_8 =$ EXHAUST$)$ $DT + 1 = 1$
→ SET K2 EXHAUST F/F
SET 1st & LAST F/F

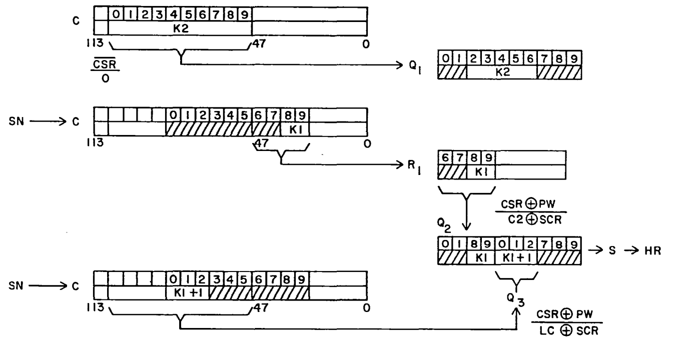
K1 { 2nd ADRS $K1 + RA \rightarrow F \rightarrow$ ADRS XMIT
$LA - LE \rightarrow LA$ $(15_8 - 12_8 = 3)$ $DT + 1 = 2$
→ TEST K1 EXHAUST
$(K1 + 1) + RA \rightarrow F \rightarrow$ ADRS XMIT $DT + 1 = 3$, $BF + 1 = 1$
$LA - LE$ $(3_8 - 12_8 =$ EXHAUST$)$
→ SET K1 EXHAUST

$\overline{K1}$ { 1st WRITE $K2 + RA \rightarrow F$ SEQ WILL WAIT UNTIL HR FULL BEFORE ADRS XMIT

MOVE EXAMPLE # 4

SHORT MOVE C1 > C2

$L = 5_8$
$C1 = 8$
$C2 = 2$



DATA SEQ - ONE WORD

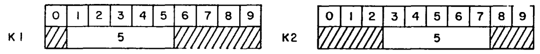
1st DATA $K2 \rightarrow CR9 \rightarrow H \rightarrow C \xrightarrow[0]{CSR} Q_1$
$DT - 1 = 2$

2nd DATA • C1 > C2 • BF = 1 $K1 \rightarrow CR9 \rightarrow H \rightarrow C \rightarrow SN \rightarrow R_1 \frac{CSR \oplus PW}{C2 \oplus SCR} Q_2$
$DT - 1 = 1$

3rd DATA $K1 + 1 \rightarrow CR9 \rightarrow H \rightarrow C \rightarrow SN \rightarrow C \xrightarrow[LC \oplus SCR]{CSR \oplus PW} S \rightarrow HR \rightarrow$ DATA XMIT, $F \rightarrow$ ADRS XMIT
$DT - 1 = 0$

5-2-40

SOURCE   DESTINATION



INSTRUCTION DECODE SEQ

$LC \rightarrow LF$
$C2 \rightarrow LE$

START SEQ

$LE + LF \rightarrow LC = 10_8$ $(L + C2)$
$C2 - C1 \rightarrow SCR = 2_8$ $SCR \rightarrow SK$
$0 \rightarrow CSR$
$SCR \rightarrow PW = 2_8$
$LA + C1 \rightarrow LA = 6_8$
$-12_8 \rightarrow LE$

ADDRESS SEQ

$\overline{K1}$ { 1st ADRS $K2 + RA \rightarrow F \rightarrow$ ADRS XMIT
$LC - LE$ $(10_8 - 12_8 =$ EXHAUST$)$ $DT + 1 = 1$
→ SET K2 EXHAUST F/F
SET 1st & LAST F/F

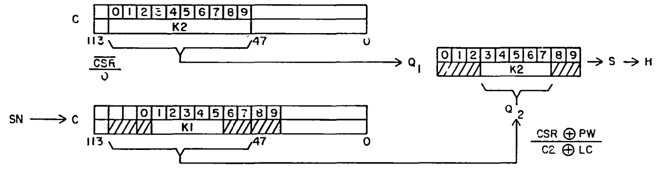
K1 { 2nd ADRS $K1 + RA \rightarrow F \rightarrow$ ADRS XMIT $DT + 1 = 2$
$LA - LE$ $(6_8 - 12_8 =$ EXHAUST$)$ $BF + 1 = 1$
→ SET K2 EXHAUST

$\overline{K1}$ { 1st WRITE $K2 + RA \rightarrow F$ SEQ WILL WAIT UNTIL HR FULL BEFORE ADRS XMIT

MOVE EXAMPLE # 5

SHORT MOVE C2 > C1

$L = 5_8$
$C1 = 1$
$C2 = 3$



DATA SEQ - ONE WORD

1st DATA $K2 \rightarrow CR9 \rightarrow H \rightarrow C \xrightarrow[0]{\overline{CSR}} Q_1$
$DT - 1 = 1$

2nd DATA • C2 > C1 $K1 \rightarrow CR9 \rightarrow H \rightarrow C \rightarrow SN \rightarrow C \xrightarrow[C2 \oplus LC]{CSR \oplus PW} Q \rightarrow S \rightarrow HR \rightarrow$ DATA XMIT, $F \rightarrow$ ADRS XMIT
$DT - 1 = 0$

5-2-41

	INSTR FLOW EX 1, EX 2 SHORT MOVE C1 > C2, C2 > C1		19981800	A
			FIGURES 5-2-40,41	5-2-84-2

TABLE 5-2-26. COMPARE/MOVE COMMAND TIMING

SEQUENCE: SHORT DATA

PAGE 1 OF 2

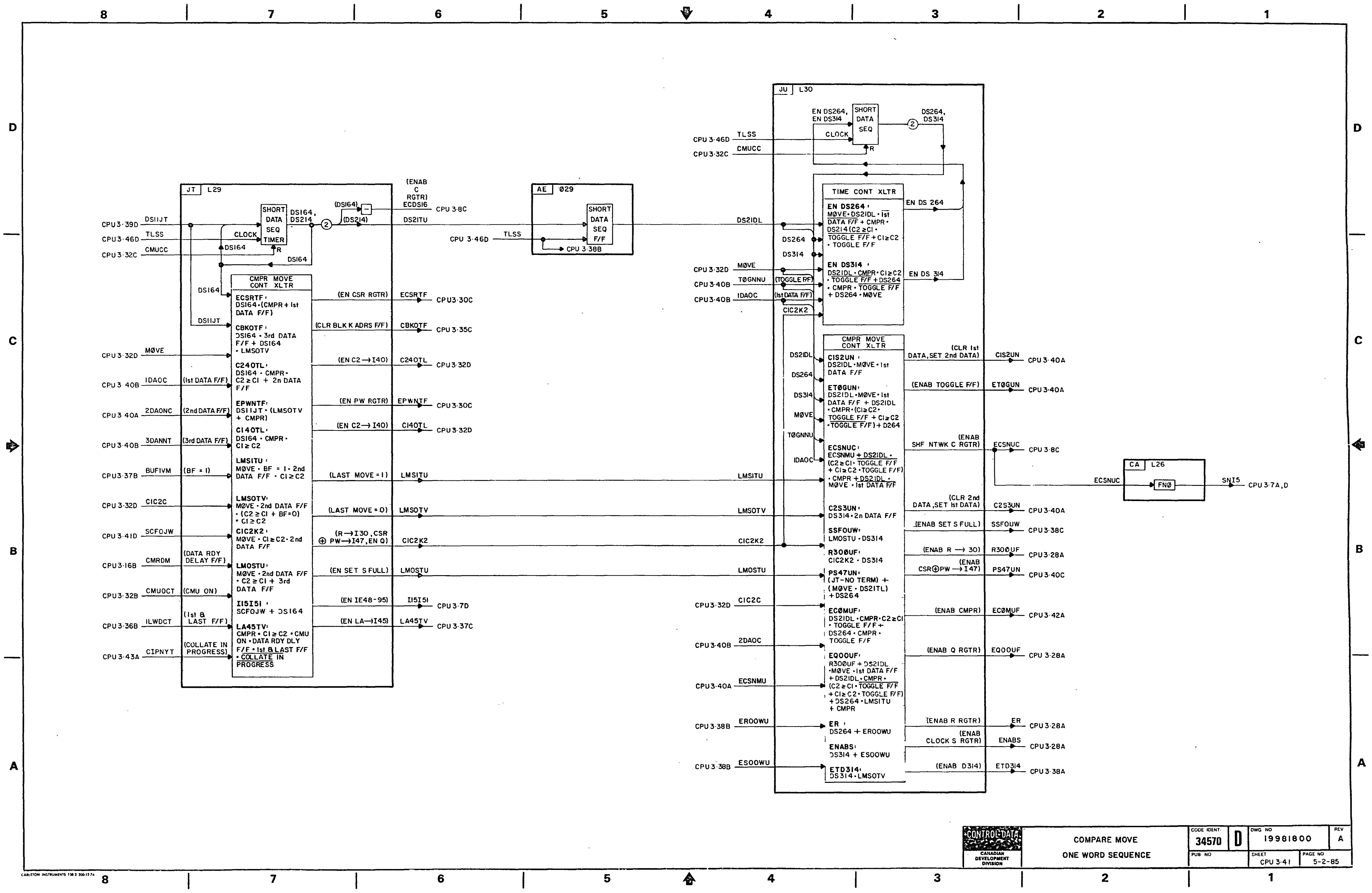
TIME	SIGNAL NAME	TEST POINT	P	COMMAND	CONDITION	COMMENTS
	CMDRJX (3.39)	M32-10	F		DATA READY DELAY FF	
DS114	ENHBH (3.39)			ENABLE H RGTR.	$\overline{\text{DATA READY DELAY FF}}$. 1ST & LAST FF . $\overline{\text{COLLATE IN PROGRESS}}$	
				DECREMENT DATA COUNTER	$\overline{\text{UPDKPJ. UPDKQJ}}$. 1ST & LAST FF . $\overline{\text{COLLATE IN PROGRESS}}$	
	G400.G401 (3.30)			SELECT I45 $\rightarrow$ I40		
	EPWNTF (3.41)			ENABLE PW RGTR	COMPARE + LMSOTV	
	DSINT (3.39)	L29-4	T	ENABLE SHORT DATA SEQ 164		
	LMS0 TV (3.41)				MOVE . 2ND DATA . [C23C1 + BUF=0 . C17C2]	LAST MOVE=0
	LMS1 TU (3.41)	L30-2			MOVE . 2ND DATA . BUF=1 . C13C2	LAST MOVE=1
DS164	F46X (3.8)			SELECT H $\rightarrow$ I15		
	I15I51 (3.7)			SELECT I15 $\rightarrow$ I5		
	ECDS16 (3.41)	I25-1	F	ENABLE C RGTR		
	$\overline{\text{G450}}$.G451 (3.31)			SELECT LC $\rightarrow$ I45		$\overline{\text{LA} \rightarrow \text{I45}}$
	ECSRTF (3.41)	N25-12	F	ENABLE CSR RGTR	COMPARE + $\overline{\text{1ST DATA FF}}$	
	C240TL (3.41)	M26-7	F	SELECT C2 $\rightarrow$ I40	COMPARE . C2 $\geq$ C1 + 2ND DATA FF	
	C140TL (3.41)	M26-12	F	SELECT C1 $\rightarrow$ I40	COMPARE . C1 $>$ C2	
	$\overline{\text{C140TL}}$. $\overline{\text{C240TL}}$ (3.41)			SELECT I45 $\rightarrow$ I40		3RD DATA
	CBKOTF (3.41)			CLR BLOCK K ADRS FF	LMOSTV + 3RD DATA	
				ENABLE SHORT DATA SEQ 214	DS164	

TABLE 5-2-26. COMPARE/MOVE COMMAND TIMING

SEQUENCE: SHORT DATA (cont.)

PAGE 2 OF 2

TIME	SIGNAL NAME	TEST POINT	P	COMMAND	CONDITION	COMMENTS
DS214						
	$\overline{\text{G302}}$ (3.28)			SELECT C → I30		R → I30
	EQ0OUF (3.41)	L25-6	F	ENABLE Q RGTR	MOVE . 1ST DATA + [COMPARE . (C2 ≥ C1 . TOGGLE + C1> C2 . $\overline{\text{TOGGLE}}$)]	
	ETOGUN (3.41)	M29-14	F	ENABLE TOGGLE FF	MOVE . 1ST DATA + [COMPARE . (C2 ≥ C1 . TOGGLE + C1 > C2 . $\overline{\text{TOGGLE}}$)]	
	ECSNUC (3.41)	L26-10	F	ENABLE C RGTR	MOVE . $\overline{\text{1ST DATA}}$ + COMPARE . (C2 ≥ C1 . $\overline{\text{TOGGLE}}$ + C1 > C2 . TOGGLE)	
	ECSNUC (3.41)	L26-10	F	SELECT SN → I5	MOVE . 1ST DATA + COMPARE . (C2 ≥ C1 . $\overline{\text{TOGGLE}}$ + C1> C2. TOGGLE)	
	PS47UN (3.41)	M29-3	F	SELECT CSR ≠ PW → I47	COMPARE	
	$\overline{\text{PS47UN}}$ (3.41)	M29-3	T	SELECT $\overline{\text{CSR}}$ → I47	MOVE	
	ECOMUF (3.41)	N33-6	F	ENABLE COMPARE SEQ	COMPARE . C2 ≥ C1 . TOGGLE	
				ENABLE SHORT DATA-DS314	COMPARE . C1> C2 . TOGGLE	
	C1S2UN (3.41)	M29-12	F	CLR 1ST DATA FF	MOVE . 1ST DATA	
	C1S2UN (3.41)	M29-12	F	SET 2ND DATA FF	MOVE . 1ST DATA	
				ENABLE DS264	MOVE . $\overline{\text{1ST DATA}}$ + COMPARE . (C2 ≥ C1 . $\overline{\text{TOGGLE}}$ + C1> C2 . TOGGLE)	
DS264						
	ER (3.41)	M07-8	T	ENABLE R RGTR		
	$\overline{\text{G302}}$ (3.28)			SELECT C → I30		
	PS47UN (3.41)	M29-3	F	SELECT CSR ≠ PW → I47		
	ETOGUN (3.41)	M29-14	F	ENABLE TOGGLE FF		
	EQ00UF (3.41)	L25-6	F	ENABLE Q RGTR	LMSITU + COMPARE	
	ECQMUF (3.41)	N33-6	F	ENABLE COMPARE SEQ	COMPARE . TOGGLE	
	(3.41)			ENABLE SHORT DATA DS314	MOVE + COMPARE . TOGGLE	
DS314	ENABS (3.41)	M14-14	T	ENABLE S RGTR		
	SSFOUW (3.41)	M30-1	F	SET S FULL FF	MOVE . 2ND DATA . C2 ≥ C1 + 3RD DATA	
	C2S3UN (3.41)	M29-9	F	CLR 2ND DATA FF	2ND DATA	
	C2S3UN (3.41)	M29-9	F	SET 3RD DATA FF	2ND DATA	
	R30OUF (3.41)	L25.5	F	SELECT R → I30	MOVE . C1 > C2. 2ND DATA FF	
	R30OUF/ (3.41)					
	PS47UN (3.41)			SELECT CSR, ≠ PW → I47	MOVE . C1 > C2. 2ND DATA FF	
	R30OUF/ (3.41)					
	EQ00UF (3.41)			ENABLE Q RGTR	MOVE . C1 > C2. 2ND DATA FF	



JT L29
SHORT DATA SEQ TIMER
CLOCK
DS164, DS214
CPU3-39D DS11JT
CPU3-46D TLSS
CPU3-32C CMUCC
CMPR MOVE CONT XLTR
ECSRTF: DS164·(CMPR+1st DATA F/F)
CBKOTF: DS164·3rd DATA F/F+DS164·LMSOTV
C240TL: DS164·CMPR·C2≥C1 + 2n DATA F/F
EPWNTF: DS11JT·(LMSOTV + CMPR)
C140TL: DS164·CMPR·C1≥C2
LMSITU: MØVE·BF=1·2nd DATA F/F·C1≥C2
LMSOTV: MØVE·2nd DATA F/F·(C2≥C1 + BF=0)·C1≥C2
C1C2K2: MØVE·C1≥C2·2nd DATA F/F
LMOSTU: MØVE·2nd DATA F/F·C2≥C1 + 3rd DATA F/F
I15I51: SCFOJW + DS164
LA45TV: CMPR·C1≥C2·CMU ON·DATA RDY DLY F/F·1st & LAST F/F·COLLATE IN PROGRESS
CPU3-32D MØVE
CPU3-40B IDAOC (1st DATA F/F)
CPU3-40A 2DAONC (2nd DATA F/F)
CPU3-40B 3DANNT (3rd DATA F/F)
CPU3-37B BUFIVM (BF=1)
CPU3-32D C1C2C
CPU3-41D SCFOJW
CPU3-16B CMRDM (DATA RDY DELAY F/F)
CPU3-32B CMUOCT (CMU ON)
CPU3-36B ILWDCT (1st & LAST F/F)
CPU3-43A CIPNYT (COLLATE IN PROGRESS)
(ENAB C RGTR) ECDS16 CPU3-8C
DS21TU
(EN CSR RGTR) ECSRTF CPU3-30C
(CLR BLK K ADRS F/F) CBKOTF CPU3-35C
(EN C2→I40) C240TL CPU3-32D
(EN PW RGTR) EPWNTF CPU3-30C
(EN C2→I40) C140TL CPU3-32D
(LAST MOVE=1) LMSITU
(LAST MOVE=0) LMSOTV
(R→I30, CSR ⊕ PW→I47, EN Q) C1C2K2
(EN SET S FULL) LMOSTU
(EN IE48-95) I15I51 CPU3-7D
(EN LA→I45) LA45TV CPU3-37C
AE Ø29
SHORT DATA SEQ F/F
CPU 3-46D TLSS
CPU 3-38B
JU L30
EN DS264, EN DS314
SHORT DATA SEQ
DS264, DS314
CPU3-46D TLSS
CPU3-32C CMUCC
TIME CONT XLTR
EN DS264: MØVE·DS21DL·1st DATA F/F + CMPR·DS214(C2≥C1·TOGGLE F/F + C1≥C2·TOGGLE F/F
EN DS314: DS21DL·CMPR·C1≥C2·TOGGLE F/F + DS264·CMPR·TOGGLE F/F + DS264·MØVE
EN DS 264
EN DS 314
DS21DL
CPU3-32D MØVE
CPU3-40B TØGNNU (TOGGLE F/F)
CPU3-40B IDAOC (1st DATA F/F)
C1C2K2
CMPR MOVE CONT XLTR
C1S2UN: DS21DL·MØVE·1st DATA F/F
ETØGUN: DS21DL·MØVE·1st DATA F/F + DS21DL·CMPR·(C1≥C2·TOGGLE F/F + C1≥C2·TOGGLE F/F)+D264
ECSNUC: ECSNMU + DS21DL·(C2≥C1·TOGGLE F/F + C1≥C2·TOGGLE F/F)·CMPR + DS21DL·MØVE·1st DATA F/F
C2S3UN: DS314·2n DATA F/F
SSFOUW: LMOSTU·DS314
R30ØUF: C1C2K2·DS314
PS47UN: (JT-NO TERM) + (MØVE·DS21TL) + DS264
ECØMUF: DS21DL·CMPR·C2≥C1·TOGGLE F/F + DS264·CMPR·TOGGLE F/F
EQ0OUF: R30ØUF + DS21DL·MØVE·1st DATA F/F + DS21DL·CMPR·(C2≥C1·TOGGLE F/F + C1≥C2·TOGGLE F/F) + DS264·LMSITU + CMPR
ER: DS264 + EROOWU
ENABS: DS314 + ESOOWU
ETD314: DS314·LMSOTV
CPU3-32D C1C2C
CPU3-40B 2DAOC
CPU3-40A ECSNMU
CPU3-38B EROOWU
CPU3-38B ESOOWU
(CLR 1st DATA, SET 2nd DATA) C1S2UN CPU3-40A
(ENAB TOGGLE F/F) ETØGUN CPU3-40A
(ENAB SHF NTWK C RGTR) ECSNUC CPU3-8C
(CLR 2nd DATA, SET 1st DATA) C2S3UN CPU3-40A
(ENAB SET S FULL) SSFOUW CPU3-38C
(ENAB R→30) R30ØUF CPU3-28A
(ENAB CSR⊕PW→I47) PS47UN CPU3-40C
(ENAB CMPR) ECØMUF CPU3-42A
(ENAB Q RGTR) EQ0OUF CPU3-28A
(ENAB R RGTR) ER CPU3-28A
(ENAB CLOCK S RGTR) ENABS CPU3-28A
(ENAB D314) ETD314 CPU3-38A
CA L26
FNØ
ECSNUC
SNI5 CPU3-7A,D

CONTROL DATA CANADIAN DEVELOPMENT DIVISION

COMPARE MOVE ONE WORD SEQUENCE

CODE IDENT		DWG NO	REV
34570	D	19981800	A
PUB NO		SHEET CPU 3-41	PAGE NO 5-2-85

CARLETON INSTRUMENTS

DETAILED PAK DIAGRAM (CPU 3.42)

COMPARE SEQUENCE

The compare sequence is enabled from either the data sequence or the collate sequence.

FROM DATA SEQUENCE

The compare sequence monitors the compare word equal signal (CWEQ) to determine the action to be performed.

Comparison Equal (CWEQ)

1. The buffer counter is decremented by two, which will allow the address sequence to initiate read requests for another pair of words.
2. The block K address FF is reset, enabling the address sequence timing chain at the next clock.
3. S full and Q full are cleared, enabling the data sequence to resume, and another compare to be initiated.

Last compare is detected by the condition (K1 exhaust and K2 exhaust) or (1st & last FF). When both K1 and K2 are exhausted for a normal compare, or 1st & last is set for a short compare, the next compare will be the last. If the result of the last comparison is an equal condition, the exit sequence will be enabled.

Comparison Unequal ($\overline{\text{CWEQ}}$)

1. The character position register (CP) is enabled, so that a code pointing to the first unequal character (from left to right) can be stored.
2. Using the CP code, the unequal character from both S and Q is stored in TS and TQ, respectively.
3. The compare sequence will enable the exit or collate sequence. The sequence enabled will depend on the instruction type being executed.

TABLE 5-2-27. COMPARE/MOVE COMMAND TIMING

SEQUENCE: COMPARE

PAGE 1 OF 2

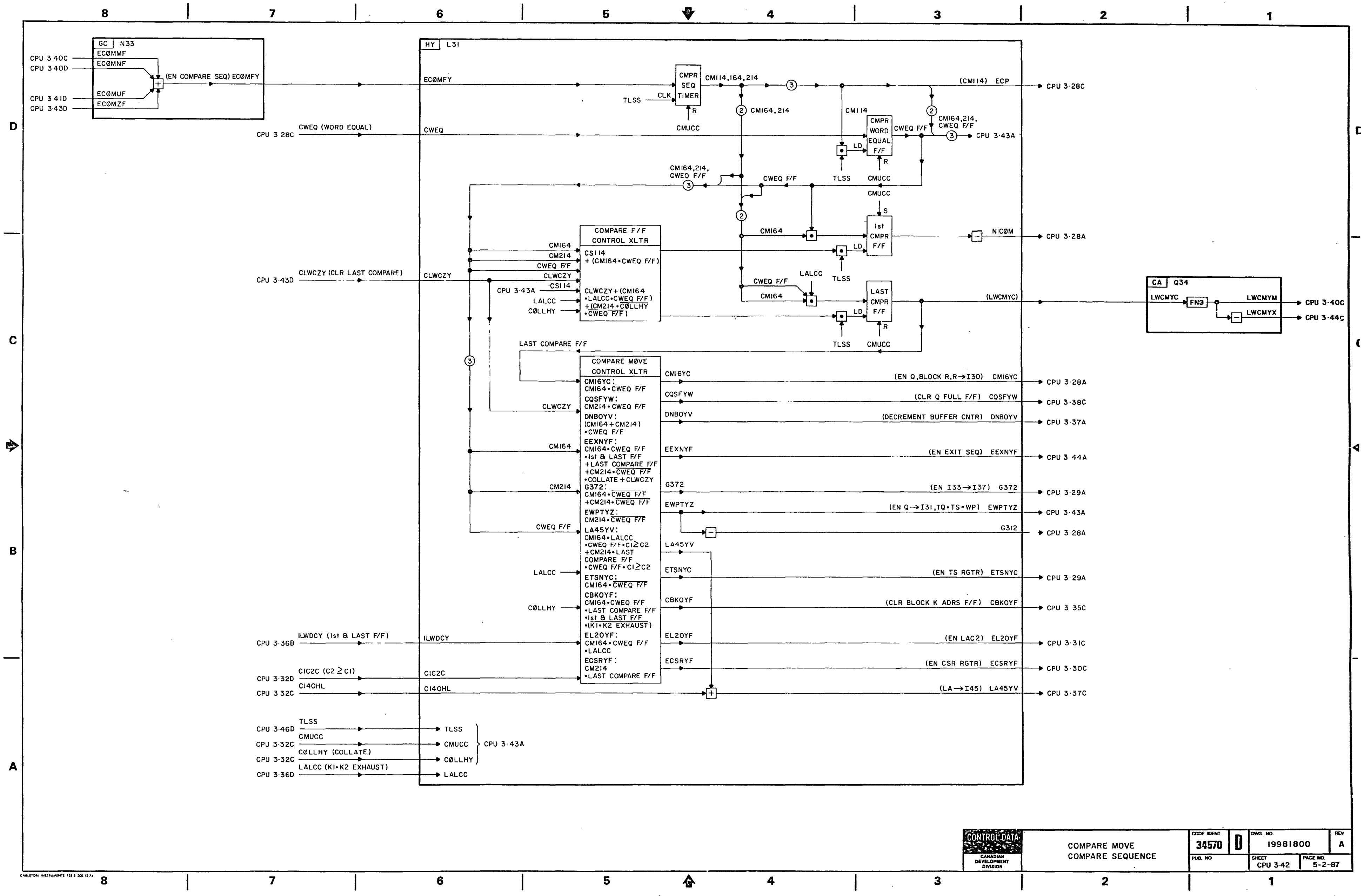
TIME	SIGNAL NAME	TEST POINT	P	COMMAND	CONDITION	COMMENTS
	ECOMFY (3.42)	N33-10	F	ENABLE COMPARE SEQ	ECOMMF + ECOMNF + ECOMUF + ECOM2F	
CM114	ECP (3.42)	L14-4	T	ENABLE CP RGTR		
	CWEQ (3.42)	L31-4	T	ENABLE CWEQ FF	CWEQ	
CM164	(3.42)			SET 1ST COMPARE FF	CWEQ FF	
	DNBOYV (3.42)	M27-13	F	DECREMENT BUFFER CNTR	CWEQ FF	
	(3.43)			CLR COLLATE IN PROGRESS FF	CWEQ FF	
	CM16YC (3.42)	N31-12	F	BLOCK R RGTR OUTPUT	CWEQ FF	
	CM16VC (3.42)	N31-12	F	SELECT R → I30	CWEQ FF	
	G470.$\overline{\text{G471}}$ (3.40)			SELECT CSR → I47		NORMALLY SELECTED
	CM16YC (3.42)	N31-12	F	ENABLE Q RGTR	CWEQ FF	
	CBKOYF (3.42)	N33-5	F	CLR BLOCK K ADRS FF	CWEQ FF . $\overline{\text{LAST COMPARE FF}}$. $\overline{\text{1ST \& LAST FF}}$. $\overline{\text{(K1 . K2 EXHAUST}}$)	
	EEXNYF (3.42)			ENABLE EXIT SEQ	CWEQ FF . LAST COMPARE FF + 1ST & LAST FF	
	(3.42)			SET LAST COMPARE FF	CWEQ FF . (K1 . K2 EXHAUST)	
	$\overline{\text{G400}}$. $\overline{\text{G401}}$ (3.30)			SELECT I45 → I40		NORMALLY SELECTED
	LA45YV (3.42)			SELECT LA → I45	CWEQ FF . C1 > C2 . (K1 . K2 EXHAUST)	
	$\overline{\text{LA45YV}}$ (3.42)			SELECT LC → I45	CWEQ FF . C2 ≥ C1 . (K1. K2 EXHAUST)	
	(3.43)			SET COLLATE IN	$\overline{\text{CWEQ FF}}$. COLLATE	
	$\overline{\text{G312}}$ (3.42)	L24-10	T	SELECT S → I31	$\overline{\text{CWEQ FF}}$	
	G372 (3.42)	M19-10	F	SELECT I33 → I37	$\overline{\text{CWEQ FF}}$	
	ETSNYC (3.42)	N14-9	F	ENABLE TS RGTR	$\overline{\text{CWEQ FF}}$	

TABLE 5-2-27. COMPARE/MOVE COMMAND TIMING

SEQUENCE: COMPARE (cont.)

PAGE 2 OF 2

TIME	SIGNAL NAME	TEST POINT	P	COMMAND	CONDITION	COMMENTS
				ENABLE CM214	CM164	
CM214						
	DNBOYV (3.42)	M27-13	F	DECREMENT BUFFER COUNTER	CWEQ FF	
	CQFSYW (3.42)	M30-7	F	CLR S FULL FF	CWEQ FF	
	CQFSYW (3.42)	M30-7	F	CLR Q FULL FF	CWEQ FF	
	G312 (3.42)	L24-10	F	SELECT Q → I31	$\overline{\text{CWEQ FF}}$	
	G372 (3.42)	M19-10	F	SELECT I33 → I37	$\overline{\text{CWEQ FF}}$	
	G312 (3.42)	L24-10	F	ENABLE TQ RGTR	$\overline{\text{CWEQ FF}}$	
	EPWTYZ (3.42)	L27-2	T	ENABLE TS = WP FF	$\overline{\text{CWEQ FF}}$	
	(3.43)			ENABLE COLLATE CS114	$\overline{\text{CWEQ FF}}$. COLLATE	
	EEXNYF (3.42)			ENABLE EXIT SEQUENCE	$\overline{\text{CWEQ FF}}$. $\overline{\text{COLLATE}}$	
				CLR LAST COMPARE FF	$\overline{\text{CWEQ FF}}$. $\overline{\text{COLLATE}}$	
	LA45YV (3.42)			SELECT LA → I45	CWEQ FF . LAST COMPARE FF . C1 > C2	
	$\overline{\text{G400}}$.$\overline{\text{G401}}$ (3.30)			SELECT I45 → I40		NORMALLY SELECTED
	ECSRYF (3.42)			ENABLE CSR RGTR	LAST COMPARE FF	
	LA45YV (3.42)			SELECT LC → I45	CWEQ FF . LAST COMPARE FF . C2 ≥ C1	



GC N33
CPU 3·40C ECØMMF
CPU 3·40D ECØMNF
CPU 3·41D ECØMUF
CPU 3·43D ECØMZF
(EN COMPARE SEQ) ECØMFY
HY L31
ECØMFY
CMPR SEQ TIMER
CLK
TLSS
R
CMUCC
CM114,164,214
(CM114) ECP
CPU 3·28C
CM164,214
CM114
CMPR WORD EQUAL F/F
LD
CWEQ F/F
CM164,214, CWEQ F/F
CPU 3·43A
CPU 3 28C CWEQ (WORD EQUAL)
CWEQ
CM164,214, CWEQ F/F
CWEQ F/F
TLSS
CMUCC
S
1st CMPR F/F
CM164
NICØM
CPU 3·28A
COMPARE F/F CONTROL XLTR
CS114 + (CM164·CWEQ F/F)
CLWCZY+(CM164 ·LALCC·CWEQ F/F) +(CM214·CØLLHY ·CWEQ F/F)
CM164
CM214
CWEQ F/F
CLWCZY
CS114
CPU 3·43A
LALCC
CØLLHY
CPU 3·43D CLWCZY (CLR LAST COMPARE)
CLWCZY
CWEQ F/F
LALCC
CM164
LAST CMPR F/F
(LWCMYC)
CA Q34
LWCMYC
FN3
LWCMYM
LWCMYX
CPU 3·40C
CPU 3·44C
LAST COMPARE F/F
COMPARE MØVE CONTROL XLTR
CM16YC: CM164·CWEQ F/F
CQSFYW: CM214·CWEQ F/F
DNBOYV: (CM164+CM214) ·CWEQ F/F
EEXNYF: CM164·CWEQ F/F ·1st & LAST F/F +LAST COMPARE F/F +CM214·CWEQ F/F ·COLLATE+CLWCZY
G372: CM164·CWEQ F/F +CM214·CWEQ F/F
EWPTYZ: CM214·CWEQ F/F
LA45YV: CM164·LALCC ·CWEQ F/F·C1≥C2 +CM214·LAST COMPARE F/F ·CWEQ F/F·C1≥C2
ETSNYC: CM164·CWEQ F/F
CBKOYF: CM164·CWEQ F/F ·LAST COMPARE F/F ·1st & LAST F/F ·(K1·K2 EXHAUST)
EL2OYF: CM164·CWEQ F/F ·LALCC
ECSRYF: CM214 ·LAST COMPARE F/F
CM16YC (EN Q,BLOCK R,R→I30) CM16YC CPU 3·28A
CQSFYW (CLR Q FULL F/F) CQSFYW CPU 3·38C
DNBOYV (DECREMENT BUFFER CNTR) DNBOYV CPU 3·37A
EEXNYF (EN EXIT SEQ) EEXNYF CPU 3·44A
G372 (EN I33→I37) G372 CPU 3·29A
EWPTYZ (EN Q→I31,TQ·TS=WP) EWPTYZ CPU 3·43A
G312 CPU 3·28A
LA45YV
ETSNYC (EN TS RGTR) ETSNYC CPU 3·29A
CBKOYF (CLR BLOCK K ADRS F/F) CBKOYF CPU 3·35C
EL2OYF (EN LAC2) EL2OYF CPU 3·31C
ECSRYF (EN CSR RGTR) ECSRYF CPU 3·30C
(LA→I45) LA45YV CPU 3·37C
CLWCZY
CM164
CM214
CWEQ F/F
LALCC
CØLLHY
CPU 3·36B ILWDCY (1st & LAST F/F) ILWDCY
CPU 3·32D C1C2C (C2≥C1) C1C2C
CPU 3·32C C14OHL C14OHL
CPU 3·46D TLSS TLSS
CPU 3·32C CMUCC CMUCC
CPU 3·32C CØLLHY (COLLATE) CØLLHY
CPU 3·36D LALCC (K1·K2 EXHAUST) LALCC
CPU 3·43A
8 7 6 5 4 3 2 1
D C B A

CONTROL DATA
CANADIAN DEVELOPMENT DIVISION

COMPARE MOVE
COMPARE SEQUENCE

CODE IDENT.		DWG. NO.	REV
34570	D	19981800	A
PUB. NO		SHEET CPU 3·42	PAGE NO. 5-2-87

CARLETON INSTRUMENTS 138'S 200-12-74

COLLATE SEQUENCE

The collate sequence is enabled from the compare sequence if a compare word unequal is detected during a 466 instruction.

The collate sequence can be divided into two sections, where timing chain sequences CS114, CS164 and CS214 form collate I, and CS264 forms collate II. (Refer to figure 5-2-42.)

COLLATE I

Timing chain sequences CS114 and CS164 are enabled by compare word unequal ($\overline{\text{CWEQ}}$) from the compare sequence, CM214. The remaining timing chain FF, CS214 is set by require address FF.

Three control FFs are conditioned by CS114 and CS164. They are: address 2, data 2 and require address.

The require address FF allows the address sequence to be enabled by enabling CS214. During CS214, the block K address FF is reset.

The address 2 FF indicates that two passes through the address sequence must be performed, since both collate characters are located in different words.

The data 2 FF indicates that two passes are required through data sequence D164. With address 2 set, each pass occurs after data ready; with address 2 reset, both passes occur consecutively after the first data ready.

The table at right shows the possible equality detection combinations: the resultant settings of the three control FFs, the number of address requests, the passes through D164, and the final code stored in WP.

CS214

CS214 is enabled by the require address FF set during CS164. At CS214, the block K address FF is cleared, and the A 0 address FF is set. A 0 address conditions the address sequence to transmit a collate table address. The upper three bits (3-5) of TS or TQ. which selects one of the eight possible collate table words, are stored in E and WP. During the address sequence, the collate table address from the A0 register is added to the select code in E to formulate the table word address.

COLLATE II CS264

Collate II is enabled from the data sequence when both collate characters have been loaded into TS and TQ.

If both collate characters are equal, the collate character equal signal (CCEQ) is generated to enable the compare sequence; otherwise the exit sequence is enabled.

Equality Detection FFs			Collate I Control FFs			Final WP	Pass in D164	No. Adrs Requests
TQ=TS	TQ=WP	TS=WP	ADRS2	DATA2	REQ ADRS			
-	1	1	0	0	0	No change	None	None
0	0	0	1	1	1	TQ	2 passes per Data Ready	2
1	0	0	0	1	1	TQ	2 passes for Data Ready	1
-	0	1	0	0	1	TQ	1 pass for Data Ready	1
-	1	0	0	0	1	TS	1 pass for Data Ready	1

COLLATE I SEQ			COLLATE II SEQ
CS114	CS164	CS214	CS264
T $\rightarrow$ I34 $\rightarrow$ I37 $\rightarrow$ TS $\rightarrow$ I38 $\uparrow$ (IF TS = WP)	TS T $\rightarrow$ I34 $\rightarrow$ I37 $\rightarrow$ TQ $\rightarrow$ I38 $\uparrow$ (IF TQ = WP) $1 \rightarrow$ REQ ADRS F/F (IF $\overline{TQ = WP} \dagger \overline{TS = WP}$) ENABLE CS264 - IF TQ = WP $\cdot$ TS = WP $1 \rightarrow ADRS_2$ F/F $\overline{TQ = WP} \cdot \overline{TS = WP} \cdot \overline{TQ = TS}$ $1 \rightarrow DATA_2$ F/F $\overline{TQ = WP} \cdot \overline{TS = WP}$ $0 \rightarrow DATA_2$ F/F TQ = WP $\textcircled{\dagger}$ TS = WP	CLR BLOCK K ADRS F/F TS $\rightarrow$ I35 $\rightarrow$ $\quad\rightarrow$ I39 $\rightarrow$ (IF $ADRS_2 \dagger$ TS = WP) TQ $\rightarrow$ I35 $\rightarrow$ $\quad\rightarrow$ I39 $\rightarrow$ (IF $\overline{ADRS_2} \cdot \overline{TQ = WP}$)	 WP E WP E ENABLE EXIT SEQ (IF $\overline{CCEQ}$) ENABLE COMPARE SEQ (IF CCEQ)

CONTROL DATA CANADIAN DEVELOPMENT DIVISION

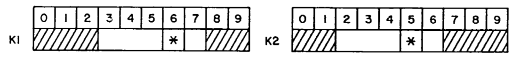
COLLATE SEQ

CODE IDENT		DWG NO	REV
34570	D	19981800	A
FIGURE 5-2-42		PAGE NO 5-2-88-1	

CARLETON INSTRUMENTS 138 3 200-12-74

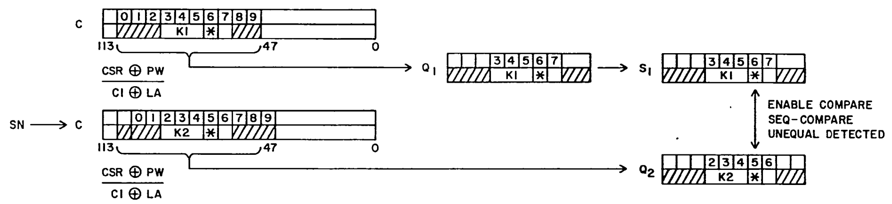
COMPARE – COLLATE   EXAMPLE #1 $C1 \geq C2$ SHORT COMPARE

$L = 5_8$
$C1 = 3$
$C2 = 2$



* NOTE: THIS EXAMPLE ASSUMES INDICATED CHARACTER OF K1 AND K2 TO BE UNEQUAL

UNEQUAL CHARACTER OF K1 = 63_8
UNEQUAL CHARACTER OF K2 = 31_8



COLLATE TABLE

	0	1	2	3	4	5	6	7
A0								
A3		37						
A6				37				

INSTRUCTION DECODE SEQ

$LA \rightarrow LF$
$C1 \rightarrow LE$

START SEQ

$LE + LF \rightarrow LA = 10_8$ (L + C1)
$C1 - C2 \rightarrow SCR = 1$ $SCR \rightarrow SK1$

$SCR \xrightarrow[]{0 \quad CSR} PW = 1$

$LC + C2 \rightarrow LC = 7_8$

ADDRESS SEQ

K1
1st ADRS $K1 + RA \rightarrow F \rightarrow$ ADRS XMIT   DT + 1 = 1, BF + 1 = 1
$LA - LE$ ($10_8 - 12_8$ = EXHAUST)
→ SET K1 EXHAUST
→ $LAC1 = 10_8$

$\overline{K1}$
2nd ADRS $K2 + RA \rightarrow F \rightarrow$ ADRS XMIT   DT + 1 = 2, BF + 1 = 2
$LC - LE$ ($7_8 - 12_8$ = EXHAUST)
→ SET K2 EXHAUST, SET BLOCK K ADRS
SET 1st & LAST F/F
$LAC1 \rightarrow LAC2 = 10_8$

A_0 ADRS $(A_0 \rightarrow F + E) \rightarrow F + RA \rightarrow$ ADRS XMIT
CLR A_0 ADRS F/F
SET BLOCK K ADRS
CLR A_2 ADRS F/F

DATA SEQ – ONE WORD

$\frac{C1 \geq C2 \cdot}{TOGGLE}$ $K1 \rightarrow CR9 \rightarrow H \rightarrow C \xrightarrow[C1 \oplus LA]{CSR \oplus PW} Q_1 \rightarrow S_1$
DT – 1 = 1

$C1 \geq C2 \cdot$ TOGGLE $K2 \rightarrow CR9 \rightarrow H \rightarrow C \rightarrow SN \rightarrow C \xrightarrow[C1 \oplus LA]{CSR \oplus PW} Q_2$
DT – 1 = 0

COMPARE SEQ

SET COLLATE IN PROGRESS
UNEQUAL CHARACTER POSITION → CP RGTR = 6_8
SELECT UNEQUAL CHARACTER FROM S → TS RGTR = 63_8 → TS
SELECT UNEQUAL CHARACTER FROM Q → TQ RGTR = 31_8 → TQ

COLLATE TABLE POSITION
CHARACTER POSITION

	5	4	3	2	1	0
TS	6			3		
TQ	3			1		

COLLATE SEQ   CS114, CS164, CS214

$\overline{TS = WP}, \overline{TQ = WP}, \overline{TQ = TS}$

SET $DATA_2$ F/F $(\overline{TQ = WP} \cdot \overline{TS = WP})$
SET $ADRS_2$ F/F $(\overline{TQ = WP} \cdot \overline{TS = WP} \cdot \overline{TQ = TS})$

SET REQUIRE ADRS F/F $(\overline{TQ = WP} \cdot \overline{TS = WP})$
CLR 1st COMPARE F/F
CLR BLOCK K ADRS F/F
SET A_0 ADRS F/F

RTS 3-5 → WP
→ E RGTR (6_8)

ADRS SEQ WILL START WITH TS CONTENT IN E RGTR

DATA SEQ

COLLATE IN PROGRESS • DT = 0 $CR9 \rightarrow T \rightarrow I34 \rightarrow I37 \rightarrow TS$
TS 0-2 ↗
(37_8)
CLR $DATA_2$ F/F
CLR T FULL F/F $(\overline{TQ = TS})$

COLLATE SEQ   CS214   (REQUIRE ADRS F/F • BLOCK K ADRS)

CLR BLOCK K ADRS F/F
SET A_0 ADRS F/F

CLR REQUIRE ADRS F/F   ($ADRS_2$ F/F)

RTQ 3-5 → WP
→ E RGTR (3_8)

ADRS SEQ WILL START WITH TQ CONTENT IN E RGTR

DATA SEQ

COLLATE IN PROGRESS • DT = 0 $CR9 \rightarrow T \rightarrow I34 \rightarrow I37 \rightarrow TQ$
TQ 0-2 ↗
(37_8)
CLR T FULL F/F
ENABLE COLLATE CS264

COLLATE SEQ   CS264

TS = TQ   ENABLE COMPARE SEQ   (IF $\overline{TS = TQ}$, ENABLE EXIT SEQ)

COMPARE SEQ

CP RGTR + 1 → CP RGTR = 7_8 (ALLOW COMPARE S AND Q CHARACTERS 7,8,9)

ASSUMING REMAINING CHARACTERS COMPARE EQUAL → ENABLE EXIT SEQ

SET LAST COMPARE F/F

EXIT SEQ

$0 \rightarrow C$ RGTR
$C \rightarrow X_0$ RGTR

INSTR FLOW EXAMPLE COMP/COLLATE C1 > C2		19981800	A
	FIGURE 5-2-43		5-2-88-2

TABLE 5-2-28. COMPARE/MOVE COMMAND TIMING

SEQUENCE: COMPARE COLLATE

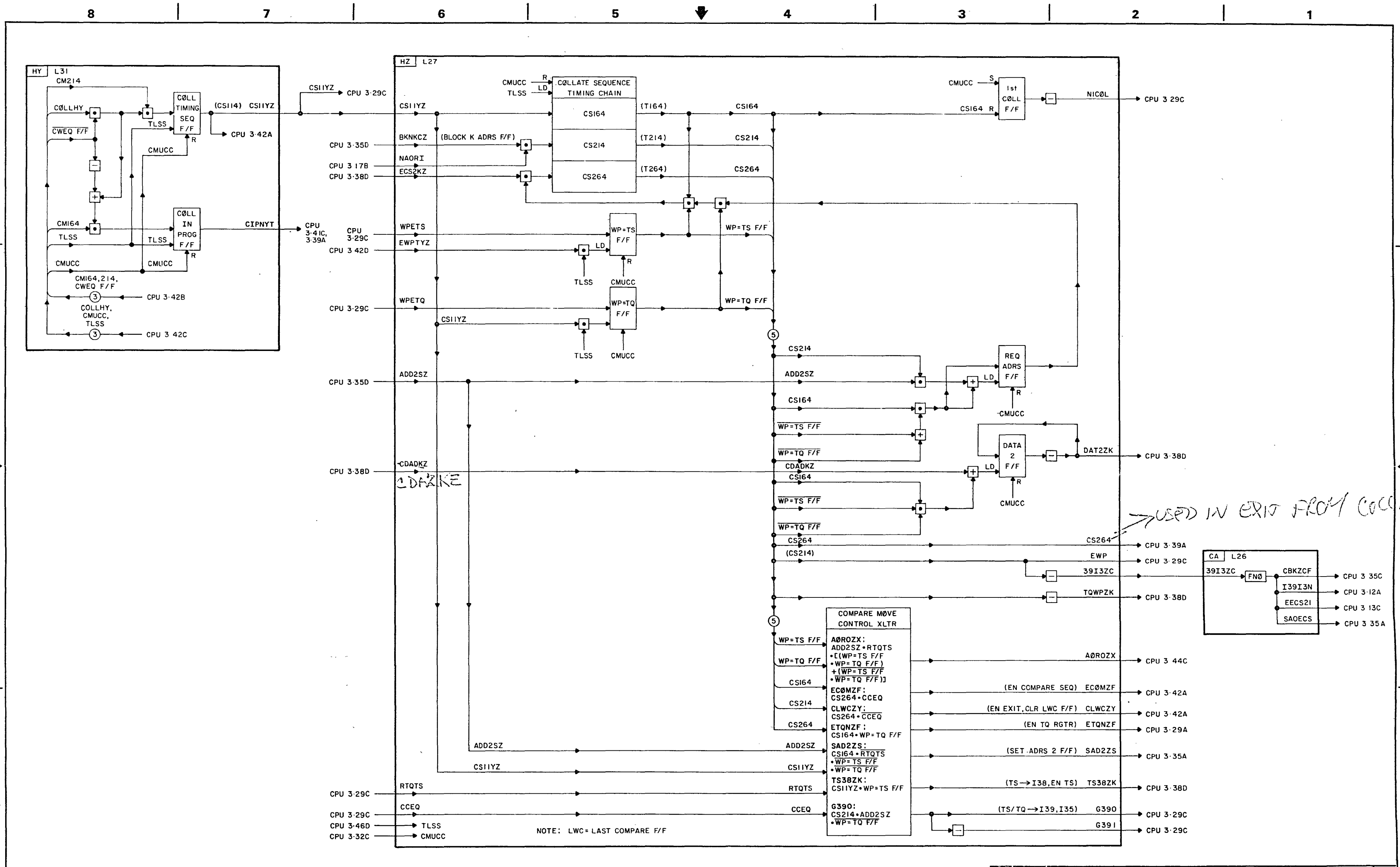
TIME	SIGNAL NAME	TEST POINT	P	COMMAND	CONDITION	COMMENTS
				SET CS114	ENABLE COLLATE - (CM214 . $\overline{\text{CWEQ FF}}$. COLLATE)	
CS114						
	(3.42)			CLR 1ST COMPARE FF		
	(3.29)			ENABLE TQ = 7S FF		
	(3.43)			ENABLE TQ = WP FF		
	TS38ZK (3.43)	M31-1	F	SELECT TS → I38	TS = WP FF	
	$\overline{\text{G372}}$ (3.42)	M19-10	T	SELECT I34→I37		
	TS38ZK (3.43)	M31-1	F	ENABLE TS RGTR	TS = WP FF	
				SET CS164	CS114	
CS164						
	$\overline{\text{TS38ZK}}$ (3.43)	M31-1	T	SELECT TQ → I38		
	$\overline{\text{G372}}$ (3.42)	M19-10	T	SELECT I34 → I37		
	ETQNZF(3.43)	L25-2	F	ENABLE TQ RGTR	TQ = WP FF	
	(3.43)			CLR 1ST COLLATE FF		
	(3.43)			ENABLE CS264	TQ = WP FF . TS = WP FF	
	(3.43)			SET DATA 2 FF	$\overline{\text{TQ = WP FF}}$. $\overline{\text{TS = WP FF}}$	
	SAD2ZS (3.43)	N28-11	F	SET ADRS 2 FF	$\overline{\text{TQ = WP FF}}$. $\overline{\text{TS = WP FF}}$. $\overline{\text{TQ = TS}}$	
				CLR ADRS 2 FF	TQ = WP FF . $\overline{\text{TS = WP FF}}$ + $\overline{\text{TQ = WP FF}}$. TS = WP FF	
				CLR DATA 2 FF	TQ = WP FF . $\overline{\text{TS = WP FF}}$ + $\overline{\text{TQ = WP FF}}$. TS = WP FF	
				SET REQUIRE ADRS FF	$\overline{\text{TQ = WP FF}}$ + $\overline{\text{TS = WP FF}}$	

TABLE 5-2-28. COMPARE/MOVE COMMAND TIMING

SEQUENCE: COMPARE COLLATE (cont.)

PAGE 2 OF 2

TIME	SIGNAL NAME	TEST POINT	P	COMMAND	CONDITION	COMMENTS
				ENABLE CS214	REQUIRE ADRS FF . BLOCK K ADRS FF . $\overline{A0RI}$	
CS214	39I3ZC (3.43)	L26-8	F	CLR BLOCK K ADRS FF		
	39I3ZC (3.43)	L26-8	F	SELECT I39 → I3		
	EPW (3.43)	L22-12	T	ENABLE WP RGTR		
	39I3ZC (3.43)	L26-8	F	ENABLE E RGTR		
	39I3ZC (3.43)	L26-8	F	SET AoADRS FF		
	(3.43)			CLR REQUIRE ADRS FF	$\overline{ADRS_2\ FF}$	ADD2SZ (2.34)
	$\overline{G390}$. G391	L23-10	F	SELECT TS → I39	$ADRS_2$ FF + $\overline{TS.WP}$	
	(3.43)	L23-8	T			
	G391	L23-10	T	TS → I35		
	G390 . $\overline{G391}$	L23-10	T	SELECT TQ → I39	$\overline{ADRS_2\ FF}$. $\overline{TQ = WP}$	
	(3.43)	L23-8	F			
		L23-10	F	TQ → I35		
				ENABLE CS264	ECS2KZ	
CS264						FROM DATA SEQ
	ECOMZF(3.43)			ENABLE COMPARE	OCEQ	
	CLWCZY(3.43)	L31-6	F	ENABLE EXIT	$\overline{CCEQ}$	
	CLWCZY(3.43)	L31-6	F	CLR LAST COMPARE FF	$\overline{CCEQ}$	



CONTROL DATA CANADIAN DEVELOPMENT DIVISION

COMPARE MOVE COLLATE SEQUENCE

CODE IDENT		DWG. NO.	REV
34570	D	19981800	A
PUB NO		SHEET CPU 3·43	PAGE NO 5-2-89

CARLETON INSTRUMENTS 1383 200-12 74

EXIT SEQUENCE

The exit sequence is enabled at the conclusion of a move or compare instruction.

MOVE INSTRUCTION

The enable exit signal (EEXNVF) is generated when: K1 and K2 are exhausted, the enable exit FF is set, and a write accept is received for the last K2 word. EEXNVF enables exit sequence E114. For a move, E114 will clear the C register only. Timing chain sequences E164 and E214 are skipped; E264 is enabled next. (Refer to figure 5-2-44.)

During E264, the contents of C, containing zero, are transferred into X0. The CMU exit signal (EMCEXH) is generated to enable the RNI sequence.

COMPARE INSTRUCTION

The enable exit signal (EEXNYF) is generated by the compare or collate sequences. A compare instruction (467) will generate enable exit if an unequal compare occurs before the last compare. It will also generate enable exit if the last compare is equal. However, the last compare FF will be set, indicating equally on the last compare.

A compare collate instruction (466) will generate enable exit if an unequal compare occurs after the appropriate collate characters are read and compared.

EXIT - COMPARE EQUAL

The normal exit for a compare equal is identical to the exit performed for a move.

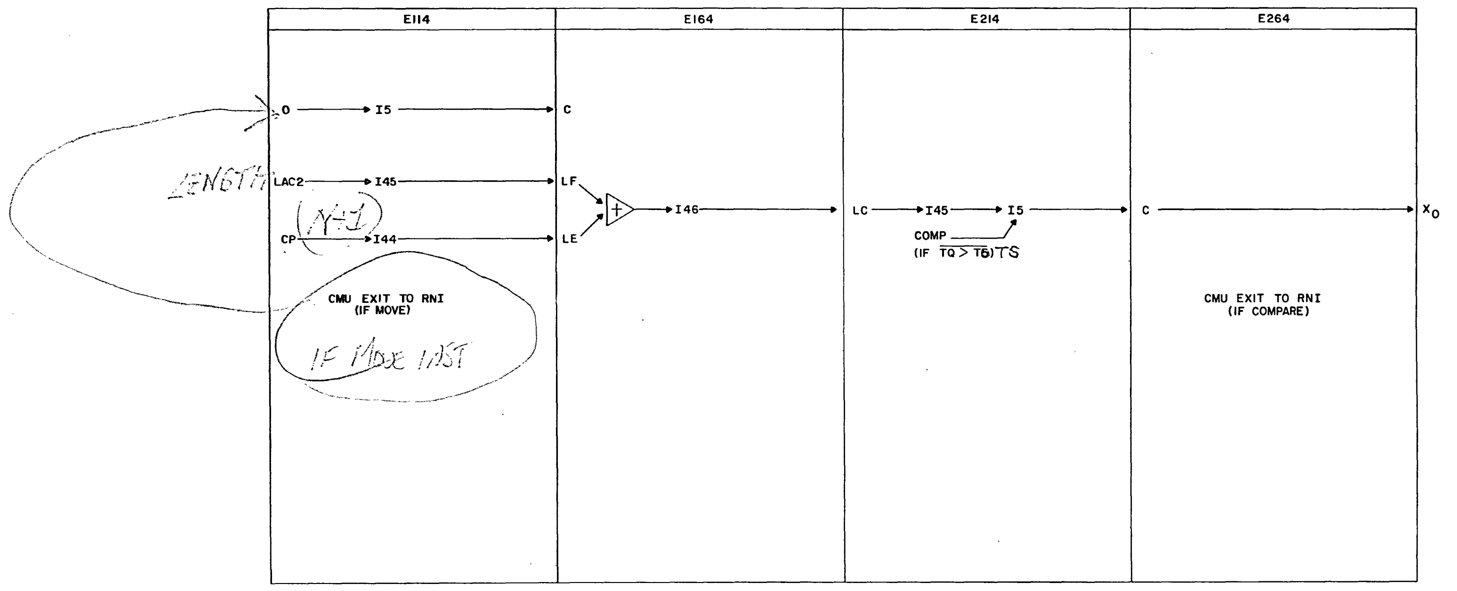
EXIT - COMPARE UNEQUAL

The exit sequence for compare unequal is used to calculate the number of characters that were not compared as the result of the unequal condition, and whether K1 is greater or less than K2. The remaining count is contained in the LAC2 register. The character position code in the CP register is subtracted from the remaining count to produce a count equal to the number of characters that have not been compared +1. The count is transferred from LC via I5 into the C register.

If $Q < S$, the C register is complemented via the I5 complement control logic. The complement of C indicates that $K1 > K2$.

If $Q > S$, the C register is not complemented. This indicates $K2 > K1$.

At E264, the count value stored in the C register is transferred into X0. The CMU exit signal (EMCEXH) is generated to enable the RNI sequence.

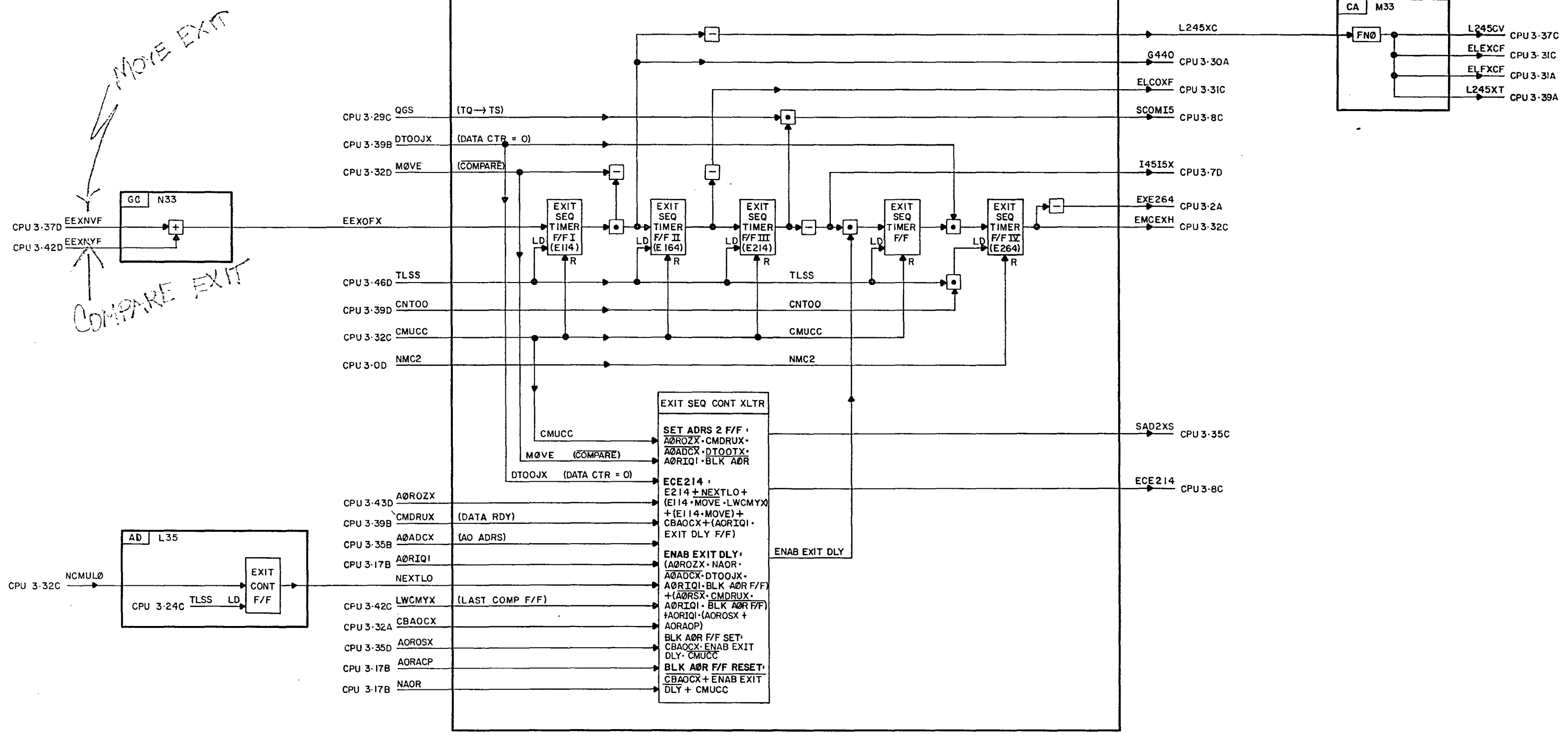


CONTROL DATA CANADIAN DEVELOPMENT DIVISION	EXIT SEQ	CODE IDENT 34570 D	DWG. NO. 19981800 REV A
		FIGURE 5-2-44	PAGE NO 5-2-90-1

TABLE 5-2-29. COMPARE/MOVE COMMAND TIMING

SEQUENCE: EXIT

TIME	SIGNAL NAME	TEST POINT	P	COMMAND	CONDITION	COMMENTS
	EEX0FX (3.44)	L34-7	T	ENABLE EXIT	EEXNVF + EEXNYF	
E114						
				SELECT 0 → I5		
	ECE214 (3.44)			ENABLE C RGTR	MOVE + COMPARE. LAST COMPARE FF	
				ENABLE E264	MOVE + COMPARE. LAST COMPARE FF	
	$\overline{\text{L245XC}}$ (3.31)	M33-10	T	SELECT LAC2 → I45	COMPARE. $\overline{\text{LAST COMPARE FF}}$	
	G440 (3.44)	N24-8	T	SELECT CP → I44	COMPARE. $\overline{\text{LAST COMPARE FF}}$	
	L245XC (3.44)	M33-10	F	ENABLE LE RGTR	COMPARE. $\overline{\text{LAST COMPARE FF}}$	
				ENABLE E164	COMPARE. $\overline{\text{LAST COMPARE FF}}$	
E164						
	G462 (3.31)			SELECT L ADDER → I46		
	ELC0XF (3.44)			ENABLE LC RGTR		
				ENABLE E214		
E214						
	$\overline{\text{L245XC}}$ (3.46)	M33-10	T	SELECT LC → I45		
	I45I5X (3.44)			SELECT I45 → I5		
	ECE214 (3.44)			ENABLE C RGTR		
	(3.44)			SET EXIT DELAY FF		
	S0OMI5 (3.44)			SELECT COMP I5	$\overline{\text{TQ > TS}}$	TQ> TS = QGS
				ENABLE EXIT 264	(EXIT DELAY FF. DT=0 + E114. COMPARE . LAST COMPARE FF + E114. MOVE + 0TOXHX) . CNT00	
E264						
	EXE264 (3.44)			SELECT X0 RGTR	$\overline{\text{BLOCK AOR FF}}$	
	EMCEXH (3.44)	L33-7	T	ENABLE CMU EXIT		
	EMCEXH (3.44)	L33-7	T	ENABLE CMU MASTER CLR		



CONTROL DATA
CANADIAN DEVELOPMENT DIVISION

COMPARE MOVE
EXIT SEQUENCE

CODE IDENT.		DWG. NO.	REV
34570	D	19981800	K
PUB. NO.		SHEET CPU 3-44	PAGE NO. 5-2-91

CARLETON INSTRUMENTS 138-3 200-12-74

DETAILED PAK DIAGRAM (CPU 3.45)

INVERTER 17
PARITY GENERATOR
OUTPUT XMITTERS

There are five sets of transmitters that allow the CPU to communicate with other units in the system.

P TRANSMITTERS - 2 SETS

The current contents of the CPU P register are continually transmitted to the two PPS chassis. This output also includes a parity bit and the condition of the run FF. The PPS can use these signals to determine abnormal CPU operation.

ECS TRANSMITTERS

The ECS coupler receives the starting address and word count from the CPU during execution of an ECS instruction. An odd parity bit (COXPAR) accompanies the transmissions. The request (COREQ) is sent to establish the start of an ECS sequence. The write signal (COWRT) will be sent with the address if a data transfer from CM to ECS is to occur. Start transfer (COSTXF) will be sent if no AOR conditions inhibit the data transfer.

CM ADDRESS TRANSMITTERS

The sequences accessing memory develop the gating for loading F into the address transmitters. The request (MEMREQ) will accompany the address if no range error exists. Parity for the address is developed as ADDPAR; however, this signal can be forced to zero by an input from the status and control register. An RNI tag accompanies requests for instructions. This is used in the CMC breakpoint test. Two control signals related to exchange jump are transmitted independently. The request exchange (CPOXRQ) signals CMC when the CPU executes a 013 instruction or an error exit exchange jump is to be made. OK exchange (CPOKX) is a response to an exchange request sent to the CPU by CMC.

CM DATA TRANSMITTERS

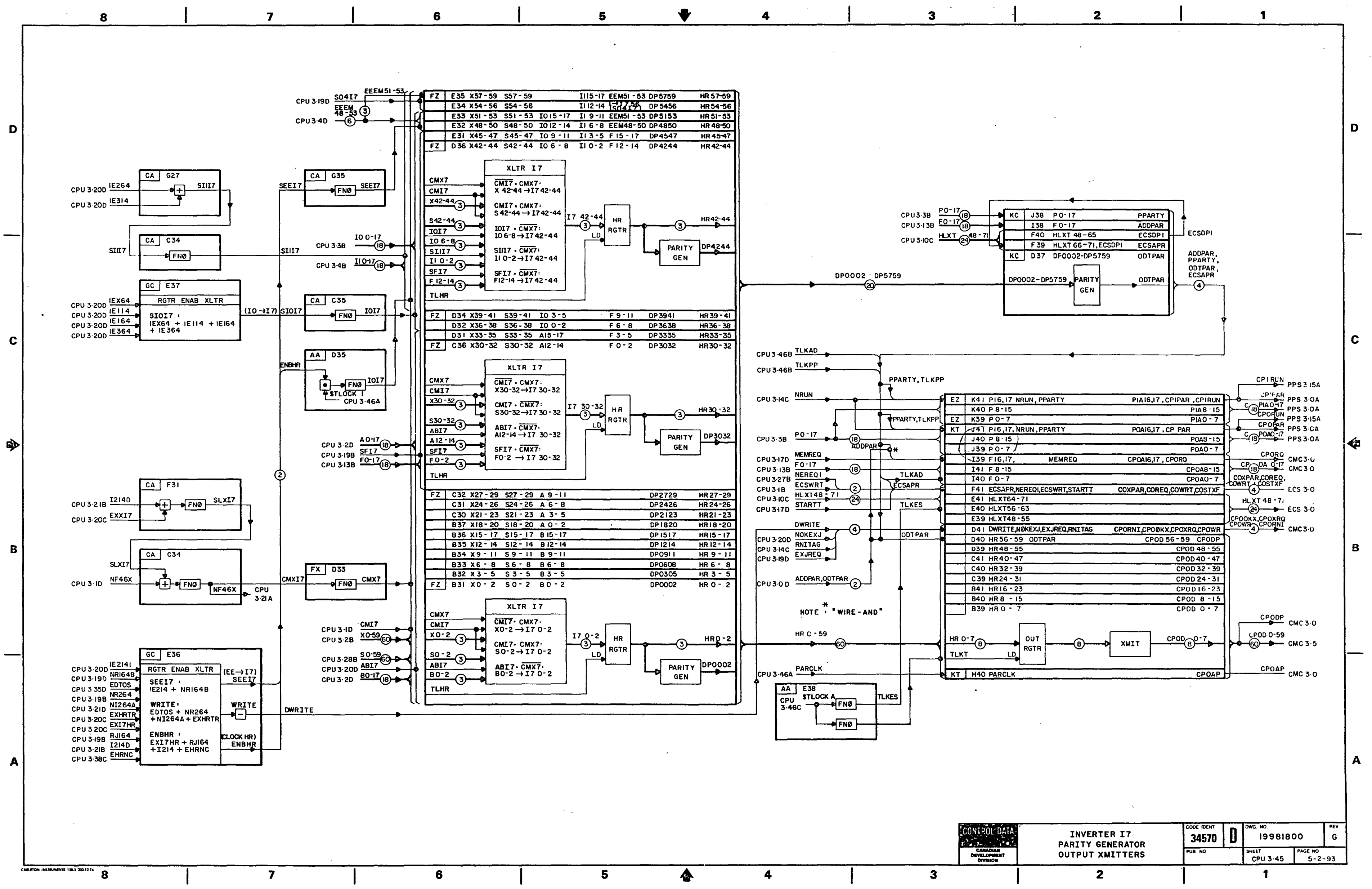
The contents of the hold register (HR) are clocked to the data transmitters continually. A single parity bit (ODTPAR) is developed to accompany data transmission. This parity can be forced to zero by the status and control register. A write signal (DWRITE) will be developed by the sequences when the data transmitters contain useful data. This signal will be transmitted to CMC as an indicator of a CM write operation.

I7 AND HOLD REGISTER

All data to be transmitted to memory is formatted in I7 and placed in the hold register. Clocking of the HR is conditioned by the sequences which store data. These sequences develop the enable HR (ENBHR) signal and the various I7 input paths. The following table illustrates the contents of HR for each sequence.

TABLE 5-2-30. CPU 3.45 KEY TEST POINTS

	FZ							KT	
BIT NO.	PAK LOC.	X (IN)	S (IN)	PAK LOC.	F (IN)	PAK LOC.	I1 (IN)	PAK LOC.	CPOD (OUT)
00	B31		12	C36		D36		B39	07
01	B31	14		C36		D36	07	B39	06
02	B31		09	C36	02	D36		B39	03
03	B32		12	D31		E31		B39	05
04	B32	14		D31		E31	07	B39	09
05	B32		09	D31	02	E31		B39	11
06	B33		12	D32		E32		B39	10
07	B33	14		D32		E32	07	B39	08
03	B33		09	D32	02	E32		B40	07
09	B34		12	D34		E33		B40	06
10	B34	14		D34		E33	07	B40	03
11	B34		09	D34	02	E33		B40	05
12	B35		12	D36		E34		B40	09
13	B35	14		D36		E34	07	B40	11
14	B35		09	D36	02	E34		B40	10
15	B36		12	E31		E35		B40	08
16	B36	14		E31		E35	07	B41	07
17	B36		09	E31	02	E35		B41	06
18	B37		12					B41	03
19	B37	14						B41	05
20	B37		09					B41	09
21	C30		12					B41	11
22	C30	14						B41	10
23	C30		09					B41	08
24	C31		12					C39	07
25	C31	14						C39	06
26	C31		09					C39	03
27	C32		12					C39	05
28	C32	14						C39	09
29	C32		09					C39	11
30	C36		12					C39	10
31	C36	14						C39	08
32	C36		09					C40	07
33	D31		12					C40	06
34	D31	14						C40	03
35	D31		09					C40	05
36	D32		12					C40	09
37	D32	14						C40	11
38	D32		09					C40	10
39	D34		12					C40	08
40	D34	14						C41	07
41	D34		09					C41	06
42	D36		12					C41	03
43	D36	14						C41	05
44	D36		09					C41	09
45	E31		12					C41	11
46	E31	14						C41	10
47	E31		09					C41	08
48	E32		12					D39	07
49	E32	14						D39	06
50	E32		09					D39	03
51	E33		12					D39	05
52	E33	14						D39	09
53	E33		09					D39	11
54	E34		12					D39	10
55	E34	14						D39	08
56	E34		09					D40	07
57	E35		12					D40	06
58	E35	14						D40	03
59	E35		09					D40	05



XLTR I7
HR RGTR
PARITY GEN
RGTR ENAB XLTR
SIOI7 = IEX64 + IE114 + IE164 + IE364
SEEI7 = IE214 + NRI64B
WRITE = EDTOS + NR264 + NI264A + EXHRTR
ENBHR = EXI7HR + RJI64 + I214 + EHRNC
CMI7 · CMX7: X 42-44 → I7 42-44
CMI7 · CMX7: S 42-44 → I7 42-44
IOI7 · CMX7: IO 6-8 → I7 42-44
SIII7 · CMX7: II 0-2 → I7 42-44
SFI7 · CMX7: FI2-14 → I7 42-44
CMI7 · CMX7: X30-32 → I7 30-32
CMI7 · CMX7: S30-32 → I7 30-32
ABI7 · CMX7: AI2-14 → I7 30-32
SFI7 · CMX7: FO-2 → I7 30-32
CMI7 · CMX7: XO-2 → I7 0-2
CMI7 · CMX7: SO-2 → I7 0-2
ABI7 · CMX7: BO-2 → I7 0-2
DP0002 - DP5759
OUT RGTR
XMIT
STLOCK A
NOTE: * "WIRE-AND"

CONTROL DATA
CANADIAN DEVELOPMENT DIVISION

INVERTER I7
PARITY GENERATOR
OUTPUT XMITTERS

CODE IDENT		DWG. NO.	REV
34570	D	19981800	G
PUB NO		SHEET CPU 3-45	PAGE NO 5-2-93

CARLETON INSTRUMENTS 138-3 200-1274

DETAILED PAK DIAGRAM (CPU 3.46)

CLOCK DISTRIBUTION

AA107-A and AD103-A

Phase 1 and phase 3 clocks are received from the master clock fanout in CMC. These signals are 50 ns apart and each has a period of 100 ns. The AB modules OR these two clocks together to produce a pulse every 50 ns. These pulses are shaped and fanned out by the AA modules to produce a CPU clock of 12 ns pulses every 50 ns.

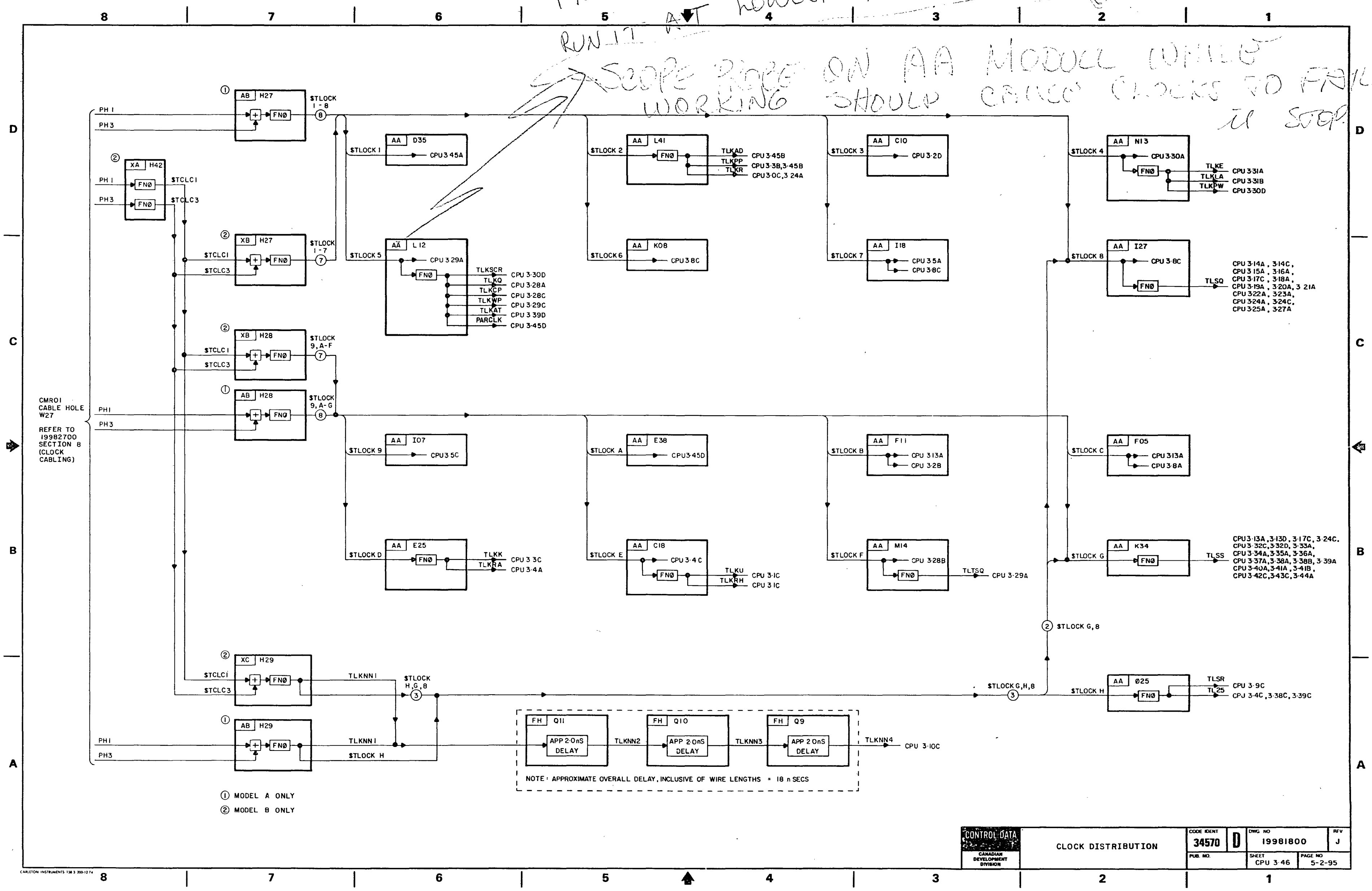
The three AB modules are aligned to produce their outputs at identical (± 1 ns) times. However, the wires leading to and from the clock fanout modules (AA) are of various lengths. This causes the clocks at the destination location to be at varying times in relation to each other. Each wire length is chosen to ensure that the most reliable transfer of signals between modules occurs.

TL25 pulses occur midway between the normal CPU clocks due to a wired inversion of the $TLOCKH signal.

AA131-B and AD105-B

The general description above is applicable except that the phase 1 and phase 3 clocks are received as differential inputs to an XA module and then fanned out to three XC modules which replace the AB modules described.

HOW TO TROUBLESHOOT MARGINAL CLOCK FAILURE
RUN IT AT LOWEST MARGIN AND ...
SCOPE PROBE ON AA MODULE WHILE WORKING SHOULD CAUSE CLOCKS TO FAIL & STOP.



AB H27
XA H42
XB H27
XB H28
AB H28
XC H29
AB H29
PH 1
PH 3
$TCLC1
$TCLC3
$TLOCK 1-8
$TLOCK 1-7
$TLOCK 9, A-F
$TLOCK 9, A-G
$TLOCK H,G,8
FNØ
AA D35 $TLOCK 1 CPU 3.45A
AA L12 $TLOCK 5 CPU 3.29A
TLKSCR CPU 3.30D
TLKQ CPU 3.28A
TLKCP CPU 3.28C
TLKWP CPU 3.29C
TLKAT CPU 3.39D
PARCLK CPU 3.45D
AA L41 $TLOCK 2
TLKAD CPU 3.45B
TLKPP CPU 3.3B, 3.45B
TLKR CPU 3.0C, 3.24A
AA K08 $TLOCK 6 CPU 3.8C
AA C10 $TLOCK 3 CPU 3.2D
AA I18 $TLOCK 7 CPU 3.5A CPU 3.8C
AA N13 $TLOCK 4 CPU 3.30A
TLKE CPU 3.31A
TLKLA CPU 3.31B
TLKPW CPU 3.30D
AA I27 $TLOCK 8 CPU 3.8C
TLSQ CPU 3.14A, 3.14C, CPU 3.15A, 3.16A, CPU 3.17C, 3.18A, CPU 3.19A, 3.20A, 3.21A, CPU 3.22A, 3.23A, CPU 3.24A, 3.24C, CPU 3.25A, 3.27A
CMR01 CABLE HOLE W27
REFER TO 19982700 SECTION 8 (CLOCK CABLING)
AA I07 $TLOCK 9 CPU 3.5C
AA E38 $TLOCK A CPU 3.45D
AA F11 $TLOCK B CPU 3.13A CPU 3.2B
AA F05 $TLOCK C CPU 3.13A CPU 3.8A
AA E25 $TLOCK D TLKK CPU 3.3C TLKNA CPU 3.4A
AA C18 $TLOCK E CPU 3.4C TLKU CPU 3.1C TLKRH CPU 3.1C
AA M14 $TLOCK F CPU 3.28B TLTSQ CPU 3.29A
AA K34 $TLOCK G TLSS CPU 3.13A, 3.13D, 3.17C, 3.24C, CPU 3.32C, 3.32D, 3.33A, CPU 3.34A, 3.35A, 3.36A, CPU 3.37A, 3.38A, 3.38B, 3.39A, CPU 3.40A, 3.41A, 3.41B, CPU 3.42C, 3.43C, 3.44A
$TLOCK G,8
$TLOCK G,H,8
AA Ø25 $TLOCK H TLSR CPU 3.9C TL25 CPU 3.4C, 3.38C, 3.39C
TLKNN1
$TLOCK H
FH Q11 APP 2.0nS DELAY
TLKNN2
FH Q10 APP 2.0nS DELAY
TLKNN3
FH Q9 APP 2.0nS DELAY
TLKNN4 CPU 3.10C
NOTE: APPROXIMATE OVERALL DELAY, INCLUSIVE OF WIRE LENGTHS = 18 n SECS
① MODEL A ONLY
② MODEL B ONLY
8 7 6 5 4 3 2 1
D C B A

CONTROL DATA CANADIAN DEVELOPMENT DIVISION

CLOCK DISTRIBUTION	CODE IDENT 34570	D	DWG NO 19981800	REV J
	PUB. NO.		SHEET CPU 3.46	PAGE NO 5-2-95

CARLETON INSTRUMENTS

SECTION 6

MAINTENANCE

SECTION 7

PARTS DATA

SECTION 8

WIRE LISTS

Information for these sections is included in separate manuals. Refer to the system publication index at the front of this manual for publication numbers.

APPENDIX A

GLOSSARY

(To be supplied later)

COMMENT SHEET

MANUAL TITLE	CDC CYBER 170 Models 172/173/174 Central Processor Unit Hardware Maintenance Manual
PUBLICATION NO.	19981800
REVISION	L

FROM: NAME:

BUSINESS ADDRESS:

CUT ALONG LINE

NO POSTAGE STAMP NECESSARY IF MAILED IN U. S. A.

FOLD ON DOTTED LINES AND STAPLE

STAPLE

STAPLE

FOLD

FOLD

FIRST CLASS
PERMIT NO. 8241

MINNEAPOLIS, MINN.

BUSINESS REPLY MAIL
NO POSTAGE STAMP NECESSARY IF MAILED IN U.S.A.

CUT ALONG LINE

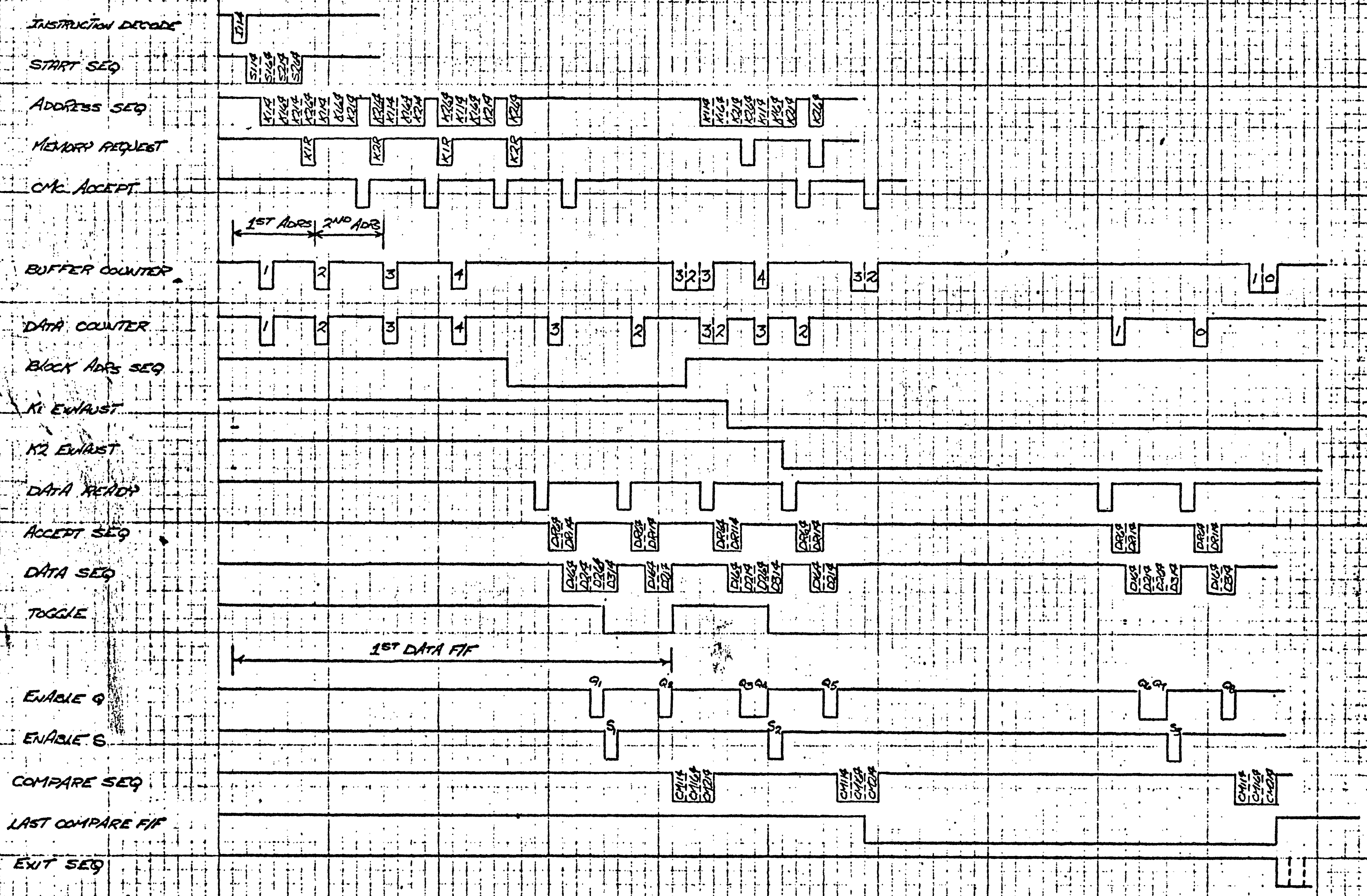
POSTAGE WILL BE PAID BY

CONTROL DATA CORPORATION
Publications and Graphics Division
ARH219
4201 North Lexington Avenue
Saint Paul, Minnesota 55112

FOLD

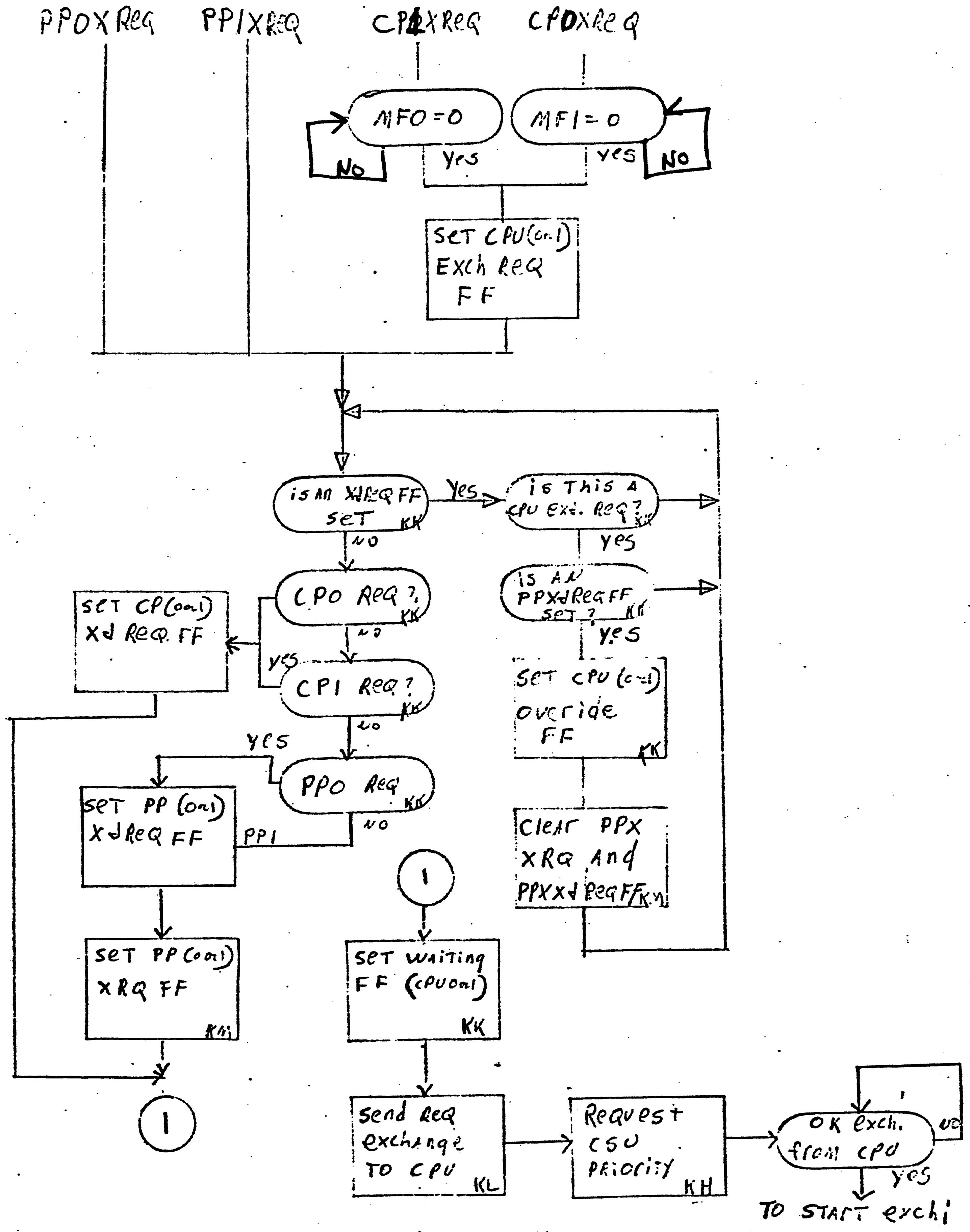
FOLD

Compare



INSTRUCTION DECODE
START SEQ
ADDRESS SEQ
MEMORY REQUEST
CMC ACCEPT
1ST ADRS
2ND ADRS
BUFFER COUNTER
DATA COUNTER
BLOCK ADRS SEQ
K1 EXHAUST
K2 EXHAUST
DATA READY
ACCEPT SEQ
DATA SEQ
TOGGLE
1ST DATA F/F
ENABLE Q
ENABLE S
COMPARE SEQ
LAST COMPARE F/F
EXIT SEQ
K1R
K2R
Q1
Q2
Q3 Q4
Q5
Q6 Q7
Q8
S1
S2
S3

Exchange jump Priority



DESCRIPTION

The single CRT CC545 display replaced the dual CRT DD60 or CC538. The data and unblank signals for A, B, C, D, etc. display is always coming over to the display from the controller. However, the data is only looked at when the switch is in the desired position.

The system software controls the display refresh rate. The system alternates i. e. 10 M sec of data for left page and 10 M sec data for right page. The system software tries to maintain 20 M sec of data for a flicker-free display. When running SMM, turn the intensity down because the diagnostics are not controlling the refresh.

PRESENTATION SWITCH

Left or Right displays an 8 x 8 inch picture. Gates the unblank for the left or right page of data, deflection unchanged.

Center position enables left and right page of data to be displayed and unblanked simultaneously with a screen size of 8 x 12 inches.

Unblank — The Presentation Switch in the center position (dual display) +20 VDC is on contact 'M'. This 20 VDC energizes relay K1 on the 4DLD PC board in C12 closing contacts enabling the unblank left, right of the 4AMD in C22 and C23.

Position Yoke — This +20VDC also energizes K2 on the 4DLD in C12. When Screen Select (left or right) page is recieved, this produces a +5 volt output from pin 5 of the 002-3 in B02. The +5VDC goes through the closed contacts of K2 to the 4DMD in A14, + X position. This enables maximum range of D/A and deflection from center of CRT. The +20VDC also energizes K3 on the4DLD in C12 for -X position.

Symbol Yoke — The +20 VDC on contact "M" also energizes K1 on the 4DND deflection summing amplifiers in A16 (-X position) and A13 (+ X position). The contacts closing parallels two resistors reducing the symbol deflection output by 25%. The K1 relay on the 4DND energizes K4 and K5 on the 4DLD in C12. This reduces the output of the 5AHD "X" symbol deflection by 25%.

The reduction of 25% is:

	"Y"	"X" Position	"Y" "X"
Left	8 in x	8 in	together = 8 x 16 -25%= 8 in x 12 in
Right	8 in x	8 in	
Center Pos.	8 in x	12 in	

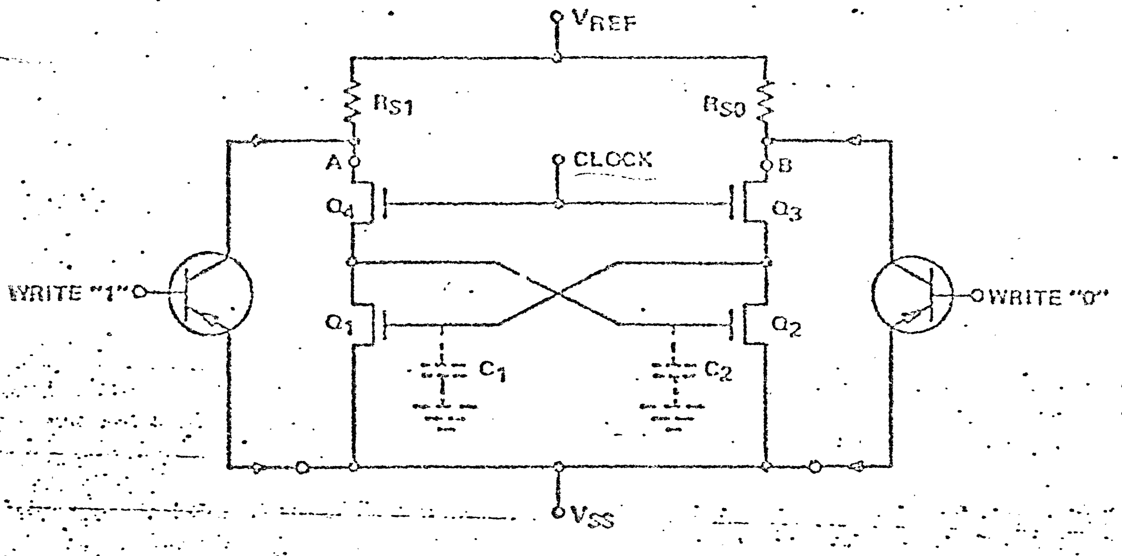


FIGURE 1. MEMORY CELL

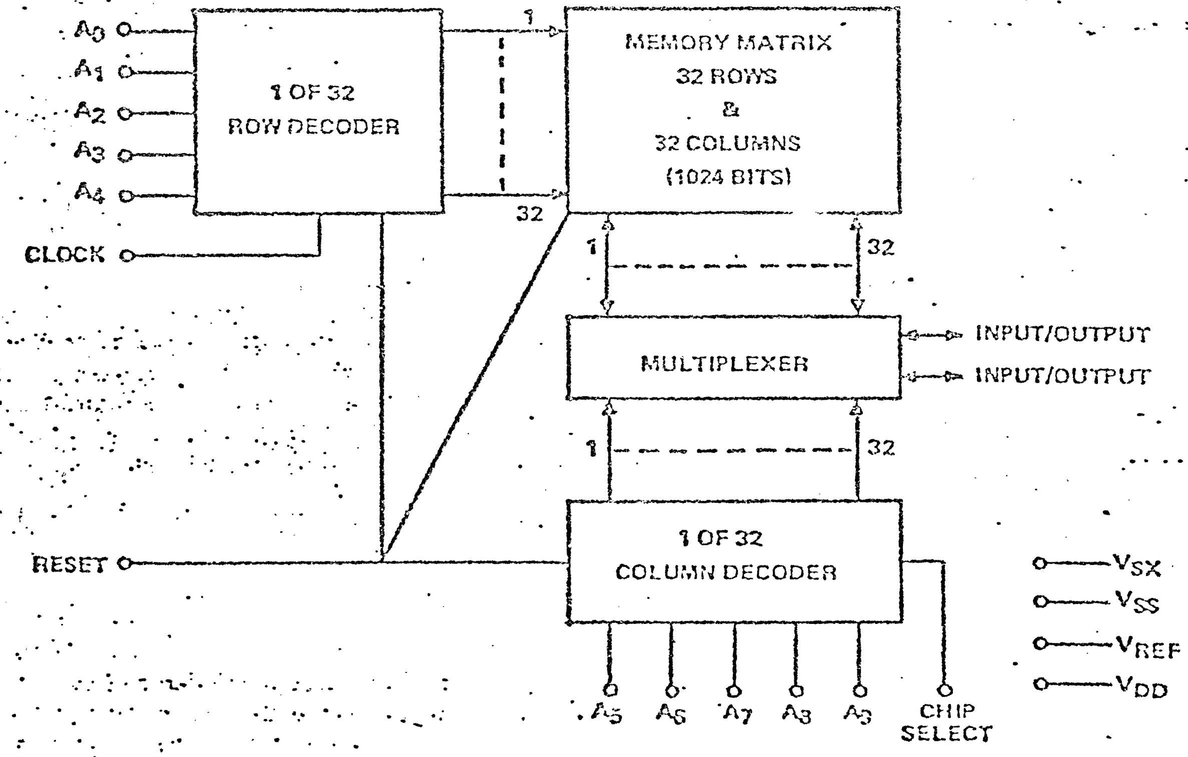
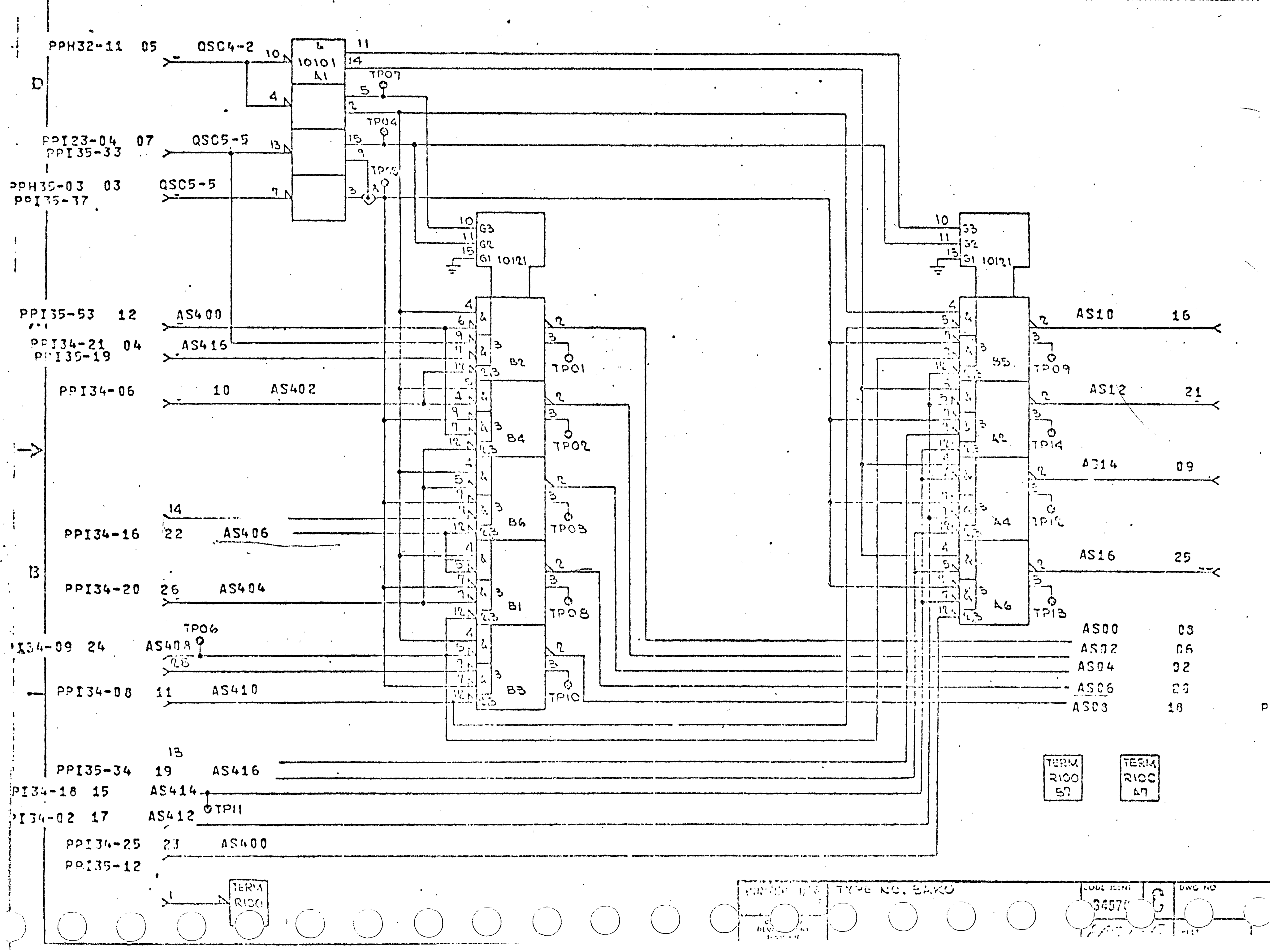


FIGURE 2. TMS 4062/63 BLOCK DIAGRAM

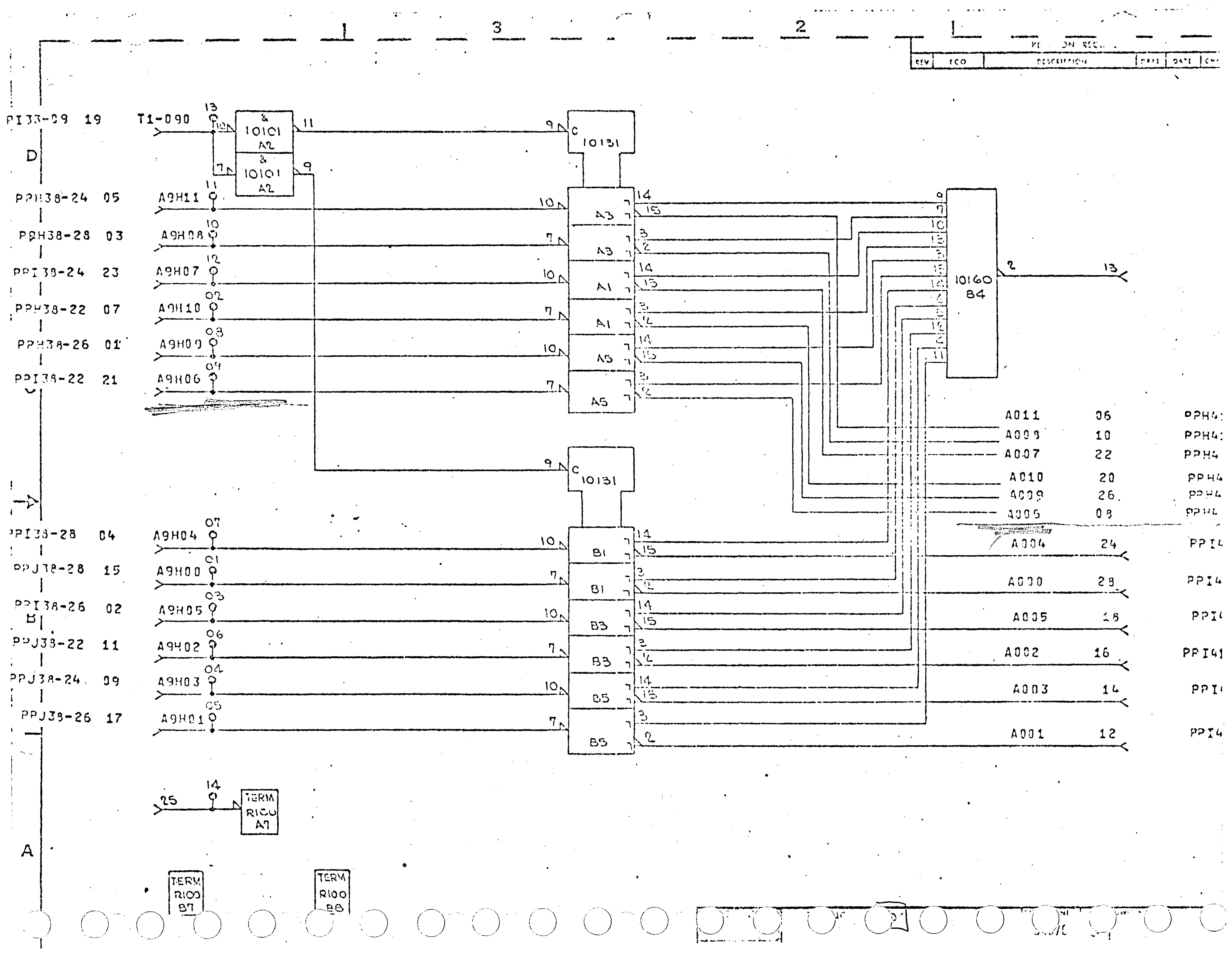


REV ECO DESCRIPTION DRFT DATE

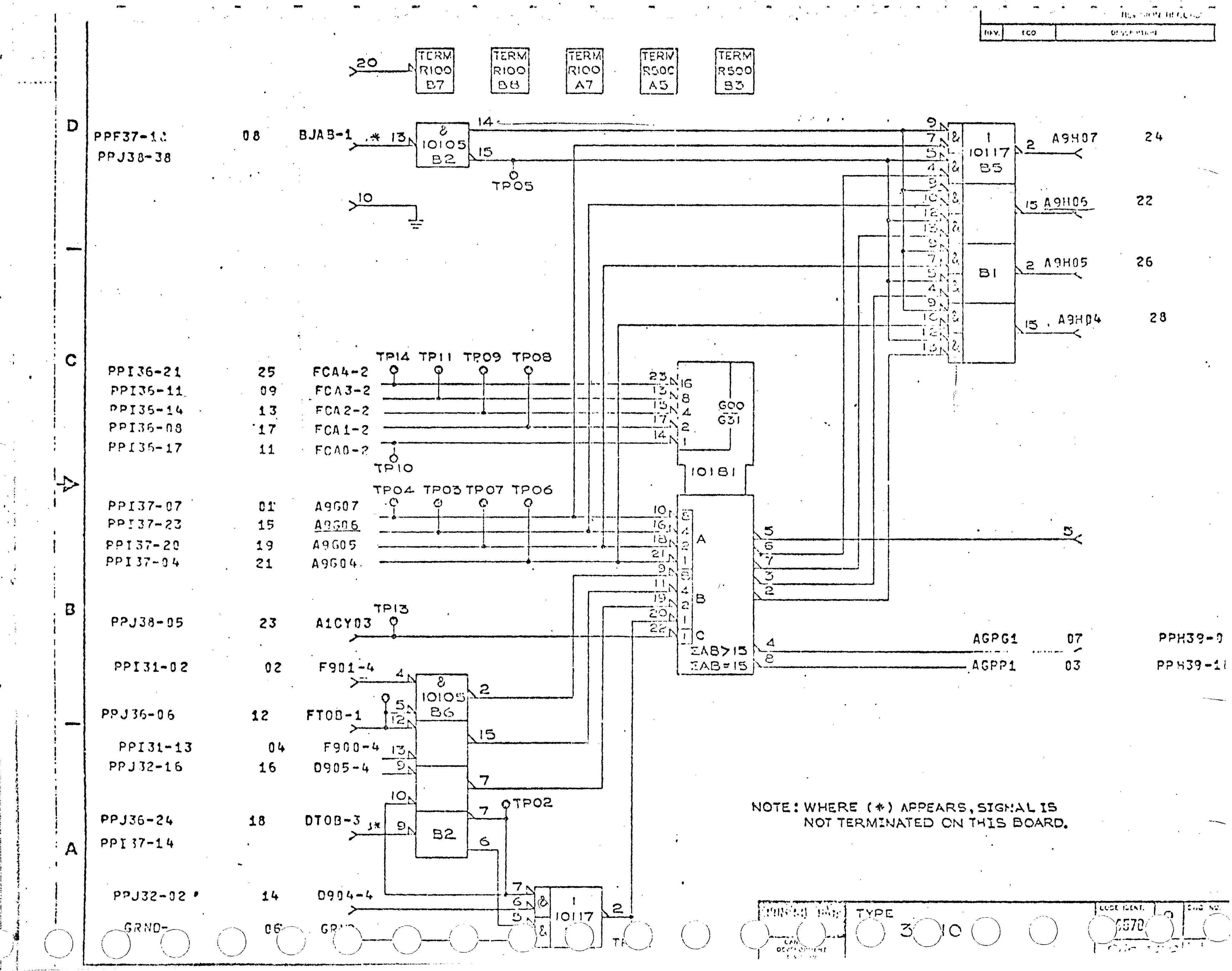
TYPE NO. SAKO

CODE IDENT 34579

DWG NO



PI33-09 19
T1-090
10101 A2
10131
PPH38-24 05
A9H11
PPH38-28 03
A9H08
PPI38-24 23
A9H07
PPH38-22 07
A9H10
PPH38-26 01
A9H09
PPI38-22 21
A9H06
A3
A1
A5
10160 B4
A011 06
A008 10
A007 22
A010 20
A009 26
A006 08
A004 24
A000 28
A005 18
A002 16
A003 14
A001 12
PPI38-28 04
A9H04
PPJ38-28 15
A9H00
PPI38-26 02
A9H05
PPJ38-22 11
A9H02
PPJ38-24 09
A9H03
PPJ38-26 17
A9H01
B1
B3
B5
TERM R100 A7
TERM R100 B7
TERM R100 B8
D
B
A



TERM R100 B7
TERM R100 B8
TERM R100 A7
TERM R50C A5
TERM R500 B3
10105 B2
10117 B5
B1
10181
G00 G31
10105 B6
10117
TP05
TP02
PPF37-12 08 BJAB-1
PPJ38-38
PPI36-21 25 FCA4-2
PPI36-11 09 FCA3-2
PPI36-14 13 FCA2-2
PPI36-08 17 FCA1-2
PPI36-17 11 FCA0-2
PPI37-07 01 A9G07
PPI37-23 15 A9G06
PPI37-20 19 A9G05
PPI37-04 21 A9G04
PPJ38-05 23 A1CY03
PPI31-02 02 F901-4
PPJ36-06 12 FT0B-1
PPI31-13 04 F900-4
PPJ32-16 16 D905-4
PPJ36-24 18 DT0B-3
PPI37-14
PPJ32-02 14 D904-4
GRND- 06 GRND
A9H07 24
A9H05 22
A9H05 26
A9H04 28
AGPG1 07 PPH39-0
AGPP1 03 PPH39-1
ΣAB>15
ΣAB=15
NOTE: WHERE (*) APPEARS, SIGNAL IS NOT TERMINATED ON THIS BOARD.
TYPE 3010

SYNDROME BIT ERROR CORRECTION

	7								6							
BIT	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48
DATA S0	-	-	-	-	-	1	1	1	-	-	1	1	1	-	-	-
S1	-	-	1	1	1	-	-	-	1	1	-	1	-	-	-	-
S2	1	1	-	1	-	-	-	-	-	-	-	1	-	-	-	-
S3	-	-	-	1	-	-	-	-	-	-	-	1	-	-	-	1
S4	-	-	-	1	-	-	-	1	1	-	-	-	1	-	1	-
S5	1	-	-	-	1	-	1	-	-	1	1	-	-	1	-	-
S6	-	1	1	-	-	1	-	-	1	1	1	1	1	1	1	1
DATA S7	1	1	1	1	1	1	1	1	-	-	-	-	-	1	1	1
	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48
BITS/COLUMN	3	3	3	<u>5</u>	3	3	3	3	3	3	3	<u>5</u>	3	3	3	3

	5								4							
BIT	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
S0	1	1	-	1	-	-	-	-	-	-	-	1	-	-	-	-
S1	-	-	-	1	-	-	-	-	-	-	-	1	-	-	-	1
S2	-	-	-	1	-	-	-	1	1	-	-	-	1	-	1	-
S3	1	-	-	-	1	-	1	-	-	1	1	-	-	1	-	-
S4	-	1	1	-	-	1	-	-	1	1	1	1	1	1	1	1
S5	1	1	1	1	1	1	1	1	-	-	-	-	-	1	1	1
S6	-	-	-	-	-	1	1	1	-	-	1	1	1	-	-	-
S7	-	-	1	1	1	-	-	-	1	1	-	1	-	-	-	-
	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
BITS/COLUMN	3	3	3	<u>5</u>	3	3	3	3	3	3	3	<u>5</u>	3	3	3	3

	3								2							
BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
S0	-	-	-	1	-	-	-	1	1	-	-	-	1	-	1	-
S1	1	-	-	-	1	-	1	-	-	1	1	-	-	1	-	-
S2	-	1	1	-	-	1	-	-	1	1	1	1	1	1	1	1
S3	1	1	1	1	1	1	1	1	-	-	-	-	-	1	1	1
S4	-	-	-	-	-	1	1	1	-	-	1	1	1	-	-	-
S5	-	-	1	1	1	-	-	-	1	1	-	1	-	-	-	-
S6	1	1	-	1	-	-	-	-	-	-	-	1	-	-	-	-
S7	-	-	-	1	-	-	-	-	-	-	-	1	-	-	-	1
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BITS/COLUMN	3	3	3	<u>5</u>	3	3	3	3	3	3	3	<u>5</u>	3	3	3	3

	1								0							
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S0	-	1	①	-	-	1	-	-	1	1	1	1	①	1	①	1
S1	1	1	①	1	①	1	①	1	-	-	-	-	-	1	①	1
S2	-	-	-	-	-	1	①	1	-	-	1	1	①	-	-	-
S3	-	-	①	1	①	-	-	-	1	1	-	1	-	-	-	-
S4	1	1	-	1	-	-	-	-	-	-	-	1	-	-	-	-
S5	-	-	-	1	-	-	-	-	-	-	-	1	-	-	-	1
S6	-	-	-	1	-	-	-	1	1	-	-	-	①	-	①	-
S7	1	-	-	-	①	-	①	-	-	1	1	-	-	1	-	-
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BITS/COLUMN	3	3	3	<u>5</u>	3	3	3	3	2	3	3	<u>5</u>	3	3	3	3

	CODE							
BIT	C7	C6	C5	C4	C3	C2	C1	C0
S0								①
S1							1	
S2						1		
S3					1			
S4				1				
S5			1					
S6		1						
S7	1							

NOTES:

IN (WRITE)
1. Code bits = "1" when number of marked bits (1) in their respective row is even.
2. Code bits are stored with data in memory.

OUT (READ)
3. Syndrome bits = "1" when number of marked bits in their respective row plus the code bit is even.
4. One Syndrome bit set implies a code bit error.
5. Three Syndrome bits set implies a data error in the column indicated by translating syndrome bits.
6. Five Syndrome bits set implies a data error in the column indicated by translating syndrome bits.
7. Any other number of Syndrome bits set implies a multiple error.

ERROR CORRECTION

5 0 1 2 3 4 6 0 0 0 0 2 0 0 0 0 0 0 1 0

	7								6								5								4							
BIT	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
S0	-	-	-	-	-	1	1	1	-	-	1	1	1	-	-	-	1	1	-	1	-	-	-	-	-	-	-	1	-	-	-	-
S1	-	-	1	1	1	-	-	-	1	1	-	1	-	-	-	-	-	-	-	1	-	-	-	-	-	-	-	1	-	-	-	1
S2	1	1	-	1	-	-	-	-	-	-	-	1	-	-	-	-	-	-	-	1	-	-	-	1	1	-	-	-	1	-	1	-
S3 0	-	-	-	1	-	-	-	-	-	-	-	1	-	-	-	1	1	-	-	-	1	-	1	-	-	1	1	-	-	1	-	-
S4	-	-	-	1	-	-	-	1	1	-	-	-	1	-	1	-	-	1	1	-	-	1	-	-	1	1	1	1	1	1	1	1
S5	1	-	-	-	1	-	1	-	-	1	1	-	-	1	-	-	1	1	1	1	1	1	1	1	-	-	-	-	-	1	1	1
S6	-	1	1	-	-	1	-	-	1	1	1	1	1	1	1	1	-	-	-	-	-	1	1	1	-	-	1	1	1	-	-	-
S7	1	1	1	1	1	1	1	1	-	-	-	-	-	1	1	1	-	-	1	1	1	-	-	-	1	1	-	1	-	-	-	-
	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
BITS/COLUMN	3	3	3	<u>5</u>	3	3	3	3	3	3	3	<u>5</u>	3	3	3	3	3	3	3	<u>5</u>	3	3	3	3	3	3	3	<u>5</u>	3	3	3	3

	3								2								1								0								CODE							
BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	C7	C6	C5	C4	C3	C2	C1	C0
S0	-	-	-	1	-	-	-	1	1	-	-	-	1	-	1	-	-	1	1	-	-	1	-	-	1	1	1	1	1	1	1	1								1
S1	1	-	-	-	1	-	1	-	-	1	1	-	-	1	-	-	1	1	1	1	1	1	1	1	-	-	-	-	-	1	1	1							1	
S2	-	1	1	-	-	1	-	-	1	1	1	1	1	1	1	1	-	-	-	-	-	1	1	1	-	-	1	1	1	-	-	-						1		
S3 0	1	1	1	1	1	1	1	1	-	-	-	-	-	1	1	1	-	-	1	1	1	-	-	-	1	1	-	1	-	-	-	-					1			
S4	-	-	-	-	-	1	1	1	-	-	1	1	1	-	-	-	1	1	-	1	-	-	-	-	-	-	-	1	-	-	-	-				1				
S5	-	-	1	1	1	-	-	-	1	1	-	1	-	-	-	-	-	-	-	1	-	-	-	-	-	-	-	1	-	-	-	1			1					
S6	1	1	-	1	-	-	-	-	-	-	-	1	-	-	-	-	-	-	-	1	-	-	-	1	1	-	-	-	1	-	1	-		1						
S7	-	-	-	1	-	-	-	-	-	-	-	1	-	-	-	1	1	-	-	-	1	-	1	-	-	1	1	-	-	1	-	-	1							
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0								
BITS/COLUMN	3	3	3	<u>5</u>	3	3	3	3	3	3	3	<u>5</u>	3	3	3	3	3	3	3	<u>5</u>	3	3	3	3	3	3	3	<u>5</u>	3	3	3	3								

NOTES:

IN (WRITE)
1. Code bits ="1" when number of marked bits (1) in their respective row is even.
2. Code bits are stored with data in memory.

OUT (READ)
3. Syndrome bits ="1" when number of marked bits in their respective row plus the code bit is even.
4. One Syndrome bit set implies a code bit error.
5. Three Syndrome bits set implies a data error in the column indicated by translating syndrome bits.
6. Five Syndrome bits set implies a data error in the column indicated by translating syndrome bits.
7. Any other number of Syndrome bits set implies a multiple error.

0

CENTRAL MEMORY

ERROR CORRECTION

	7	6	5	4	3	2	1	0	CODE
BIT	63 62 61 60 59 58 57 56	55 54 53 52 51 50 49 48	47 46 45 44 43 42 41 40	39 38 37 36 35 34 33 32	31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0	C7 C6 C5 C4 C3 C2 C1 C0
S0	- - - - - | | |	- - | | | - - -	| | - | - - - -	- - - | - - - -	- - - | - - - |	| - - - | - | -	- | | - - | - -	| | | | | | | |	|
S1	- - | | | - - -	| | - | - - - -	- - - | - - - -	- - - | - - - |	| - - - | - | -	- | | - - | - -	| | | | | | | |	- - - - - | | |	|
S2	| | - | - - - -	- - - | - - - -	- - - | - - - |	| - - - | - | -	- | | - - | - -	| | | | | | | |	- - - - - | | |	- - | | | - - -	|
S3	- - - | - - - -	- - - | - - - |	| - - - | - | -	- | | - - | - -	| | | | | | | |	- - - - - | | |	- - | | | - - -	| | - | - - - -	|
S4	- - - | - - - |	| - - - | - | -	- | | - - | - -	| | | | | | | |	- - - - - | | |	- - | | | - - -	| | - | - - - -	- - - | - - - -	|
S5	| - - - | - | -	- | | - - | - -	| | | | | | | |	- - - - - | | |	- - | | | - - -	| | - | - - - -	- - - | - - - -	- - - | - - - |	|
S6	- | | - - | - -	| | | | | | | |	- - - - - | | |	- - | | | - - -	| | - | - - - -	- - - | - - - -	- - - | - - - |	| - - - | - | -	|
S7	| | | | | | | |	- - - - - | | |	- - | | | - - -	| | - | - - - -	- - - | - - - -	- - - | - - - |	| - - - | - | -	- | | - - | - -	|
	63 62 61 60 59 58 57 56	55 54 53 52 51 50 49 48	47 46 45 44 43 42 41 40	39 38 37 36 35 34 33 32	31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0	
BITS/COLUMN	3 3 3 5 3 3 3 3	3 3 3 5 3 3 3 3	3 3 3 5 3 3 3 3	3 3 3 5 3 3 3 3	3 3 3 5 3 3 3 3	3 3 3 5 3 3 3 3	3 3 3 5 3 3 3 3	2 3 3 5 3 3 3 3	

NOTES:

IN (WRITE)
1. Code bits ="1" when number of marked bits (|) in their respective row is even.
2. Code bits are stored with data in memory.

OUT (READ)
3. Syndrome bits ="1" when number of marked bits in their respective row plus the code bit is even.
4. One Syndrome bit set implies a code bit error.
5. Three Syndrome bits set implies a data error in the column indicated by translating syndrome bits.
6. Five Syndrome bits set implies a data error in the column indicated by translating syndrome bits.
7. Any other number of Syndrome bits set implies a multiple error.

	CODE (BIN)	DATA ~~(HEX)~~			
	7 0	63			0
WRITTEN	1111 1111	0000	0000	0000	0000
READ	1111 1111	0000	0000	0000	0000
SYNDROME	0000 0000	- No Error			

OCTAL

WRITTEN	1101 1100 334₈	0000	0000	0000	0001
READ	1101 1100	0000	0000	0000	0001
SYNDROME	0000 0000	- No Error			

WRITTEN	1011 1100	0000	0000	0000	0002
READ	1011 1100	0000	0000	0000	0000
SYNDROME	0100 0011	- 1 Error, Toggle bit 1			
	1 0 3	DATA ERROR			

WRITTEN	1101 1100	0000	0000	0000	0001
READ	1101 1100	0000	0000	0000	0002
SYNDROME	1010 0011	- Double Error, No Correction			
		DATA ERROR			

WRITTEN	1101 1100	0000	0000	0000	0001
READ	1101 1000	0000	0000	0000	0001
SYNDROME	0000 0100	- 1 Error, No Correction			
		CODE BIT FAILURE			

ERROR ANALYSIS

SECDED SYNDROME CODE/CORRECTED BIT TABLE

CODE	BIT	CODE	BIT	CODE	BIT	CODE	BIT	CODE	BIT	CODE	BIT	CODE	BIT	CODE	BIT
000	NONE	040	①	100	①	140	②	200	①	240	②	300	②	340	50
001	①	041	②	101	②	141	53	201	②	241	57	301	58	341	②
002	①	042	②	102	②	142	54	202	②	242	59	302	61	342	②
003	②	043	0	103	1	143	②	203	2	243	②	303	②	343	③
004	①	044	②	104	②	144	40	204	②	244	63	304	62	344	②
005	②	045	23	105	3	145	②	205	5	245	②	305	②	345	③
006	②	046	22	106	8	146	②	206	9	246	②	306	②	346	③
007	10	047	②	107	②	147	③	207	②	247	44	307	③	347	②
010	①	050	②	110	②	150	41	210	②	250	43	310	48	350	②
011	②	051	47	111	7	151	②	211	6	251	②	311	②	351	28
012	②	052	27	112	31	152	②	212	11	252	②	312	②	352	③
013	13	053	②	113	②	153	③	213	②	253	③	313	③	353	②
014	②	054	29	114	30	154	②	214	16	254	②	314	②	354	③
015	17	055	②	115	②	155	③	215	②	255	③	315	③	355	②
016	18	056	②	116	②	156	③	216	②	256	③	316	③	356	②
017	②	057	③	117	52	157	②	217	③	257	②	317	②	357	④
020	①	060	②	120	②	160	42	220	②	260	45	320	49	360	②
021	②	061	46	121	51	161	②	221	56	261	②	321	②	361	③
022	②	062	32	122	55	162	②	222	15	262	②	322	②	362	③
023	14	063	②	123	②	163	③	223	②	263	③	323	36	363	②
024	②	064	33	124	35	164	②	224	39	264	②	324	②	364	20
025	19	065	②	125	②	165	③	225	②	265	③	325	③	365	②
026	21	066	②	126	②	166	③	226	②	266	③	326	③	366	②
027	②	067	③	127	③	167	②	227	③	267	②	327	②	367	④
030	②	070	34	130	37	170	②	230	38	270	②	330	②	370	③
031	24	071	②	131	②	171	③	231	②	271	③	331	③	371	②
032	25	072	②	132	②	172	12	232	②	272	③	332	③	372	②
033	②	073	③	133	③	173	②	233	③	273	②	333	②	373	④
034	26	074	②	134	②	174	③	234	②	274	③	334	③	374	②
035	②	075	4	135	③	175	②	235	③	275	②	335	②	375	④
036	②	076	③	136	③	176	②	236	60	276	②	336	②	376	④
037	②	077	②	137	②	177	④	237	②	277	④	337	④	377	②

NOTES:

① SYNDROME CODE BIT FAILED (SINGLE CODE BIT SET)

② DOUBLE ERROR OR MULTIPLE DOUBLE ERROR (EVEN NO. OF CODE BITS SET)

③ MULTIPLE SINGLE ERROR (THREE OR FIVE CODE BITS SET)

④ MULTIPLE ERROR (SEVEN CODE BITS SET)

THE SYNDROME CODES ABOVE ARE OCTAL REPRESENTATIONS OF THE EIGHT SYNDROME CODE BITS.